МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО
ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Факультет компьютерных наук

СБОРНИК СТУДЕНЧЕСКИХ НАУЧНЫХ РАБОТ ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ НАУК ВГУ

Выпуск 7

Под редакцией кандидата физико-математических наук
Е.А. Сирота

Издательско-полиграфический центр
Воронежского государственного университета
2013

УДК 004.65 + 004.438.5
ББК Ч 481 (2) 22
С23

Рекомендовано к печати
Ученым советом факультета компьютерных наук ВГУ

С23 **Сборник студенческих научных работ факультета компьютерных наук ВГУ** / под ред. Е.А. Сирота; Воронежский государственный университет. – Выпуск 7. – Воронеж: Издательско-полиграфический центр Воронежского государственного университета, 2013. – 160 с.
ISBN 978-5-9273-1733-2

В сборник включены научные работы студентов факультета компьютерных наук ВГУ, выполненные в 2012-2013 гг. под руководством преподавателей факультета, представленные в виде докладов и рекомендованные к опубликованию оргкомитетами студенческих научных конференций.

УДК 004.65 + 004.438.5
ББК Ч 481 (2) 22

Научное издание

СБОРНИК СТУДЕНЧЕСКИХ НАУЧНЫХ РАБОТ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ НАУК ВГУ

Выпуск 7

Подп. в печ. 9.07.2012. Формат 60x84/16. Усл. печ. л. 9,3. Тираж 80 экз. Заказ 935.
Издательско-полиграфический центр Воронежского государственного университета.
394000, г. Воронеж, пл. им. Ленина, 10. Тел. (факс): +7 (473) 259-80-26
http://www.ppc.vsu.ru; e-mail: pp_center@ppc.vsu.ru
Отпечатано с готового оригинал-макета в типографии
Издательско-полиграфического центра Воронежского государственного университета.
394000, г. Воронеж, ул. Пушкинская, 3. Тел. +7 (473) 220-41-33

ISBN 978-5-9273-1733-2

© Воронежский государственный университет, 2013
© Оформление. Издательско-полиграфический центр Воронежского государственного университета, 2013

РАСПРЕДЕЛЕННАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА VIOLA-JONES С ИСПОЛЬЗОВАНИЕМ ПРОГРАММНО-АППАРАТНОЙ АРХИТЕКТУРЫ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ CUDA В СРЕДЕ MATLAB

студ. А.В. Акимов,
проф. А.А. Сирота

Использование автоматических систем распознавания (классификации) изображений находит себе применение в самых разных областях человеческой деятельности. При этом практика зачастую требует от распознающих систем реакции в режиме реального времени. Существует множество альтернативных вариантов построения таких систем: от статистических моделей до нейронных сетей. В данной работе рассматривается метод Viola-Jones ([1] и [2]), зарекомендовавший себя во многих исследовательских работах как быстрый и надежный способ распознавания изображений. Можно выделить следующие особенности его реализации.

Получение первичных признаков классификации. На уровне первичной обработки наблюдений в исследуемой области изображения (сканирующем окне) метод осуществляет анализ разности суммарной яркости пикселей в смежных областях различной конфигурации (рис. 1), что обеспечивает формирование исходных признаков классификации или примитивов Хаара.

Рис. 1. Пример наложения примитива Хаара на участки изображения

Этот процесс имеет быструю реализацию, заключающуюся в предварительном построении интегрального изображения. Каждая его точка содержит сумму яркостей пикселей слева и сверху от нее

включительно. В итоге для расчета суммарной яркости прямоугольной области изображения в произвольном месте достаточно сделать всего четыре обращения к его интегральному представлению (по одному на каждый край этой области).

Усиление слабых классификаторов. В основе метода Viola-Jones лежит подход, называемый бустингом [3], и его модификация AdaBoost. Смысл его сводится к построению «сильного» (большая вероятность выдать верный результат) классификатора, состоящего из линейной комбинации «слабых» исходных признаков классификации (малая вероятность выдать верный результат и ориентация на конкретные ситуации).

На первом этапе обучения всем тренировочным заранее помеченным экземплярам выдаются одинаковые веса, и выбирается тот слабый алгоритм, который наберет наибольшую их сумму при классификации. Затем производится перевзвешивание так, что большие значения начинают соответствовать неверно определенным экземплярам, т.е. каждый последующий выбранный слабый классификатор специализируется на тех изображениях, с которыми не справился предыдущий.

Использование каскада классификаторов. Для ускорения производительности работы метода используется следующий подход. Итоговый классификатор строится в виде каскада описанных выше линейных комбинаций (рис. 2). При этом каждый из слоев настраивается на верное распознавание как можно большего числа объектов поиска, но учится отсеивать только те результаты, с которыми не справился предыдущий.

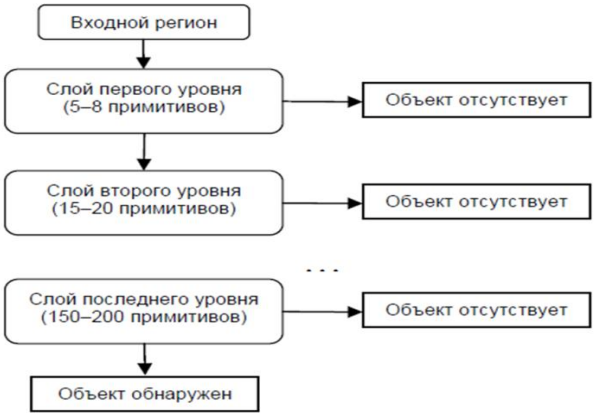

Рис. 2. Схема примера каскада классификаторов

В итоге каждый последующий слой обработки содержит все большее количество слабых классификаторов и при анализе запускается только, если предыдущий сообщает о возможности наличия в анализируемой области объекта поиска. Иначе текущее окно отбрасывается, и алгоритм переходит к следующему.

Применение обученного классификатора производится путем перемещения сканирующего окна по изображению, в том числе и в разных масштабах. При этом масштабируется классификатор, но не изображение. Для каждого из положений этого окна рассчитывается возвращаемое полученным при обучении каскадом значение и на его основе принимается соответствующее решение в виде результата классификации.

Метод характеризуется весьма длительным процессом обучения ввиду большого стартового набора слабых классификаторов, проистекающего из множества вариантов их размещения внутри сканирующего окна, и большого числа тренировочных экземпляров-изображений. Сам процесс распознавания имеет скорость, достаточную для анализа видео в реальном времени с небольшими разрешением и потенциальным числом искомых объектов (имеются ввиду уже существующие реализации).

Для увеличения производительности метода перспективным выглядит возможность использования распределенных вычислений на основе многоядерного графического процессора ввиду его дешевизны по сравнению с использованием многопроцессорных систем, обеспечивающих аналогичную производительность. Графические ускорители от разных производителей предоставляют свое собственное несовместимое друг с другом API (NVIDIA CUDA и ATI Stream), требующее некоторых специфических знаний о низкоуровневых механизмах их работы. Имеется также фреймворк OpenCL, являющийся своего рода оберткой для них и позволяющий использовать ресурсы гетерогенных систем, состоящих из множества процессоров [4].

Распределенная реализация. В данной работе для распределенной реализации алгоритма была использована технология CUDA, вызываемая из среды Matlab. Это позволило не отвлекаться на написание низкоуровневых механизмов вроде выделения памяти и ряда примитивных операций, таких как расчет интегрального изображения. Графический ускоритель представляет собой несколько мультипроцессоров, состоящих из блоков, каждый из которых можно адресовать либо одной, либо двумя координатами, и в каждом из них одновременно может функционировать несколько потоков, также идентифицируемых набором из координат (одной, двух или трех).

Выполненный анализ показывает, что наибольшего эффекта от использования графического процессора можно добиться путем запуска на нем алгоритма классификации целиком параллельно для каждого положения сканирующего окна (рисунок 1). Это положение соответствует определенным координатам потока внутри ускорителя. Сначала производятся предварительные операции, в том числе расчет интегрального изображения, а затем – для каждого масштаба анализа – параллельная обработка содержимого сканирующих окон.

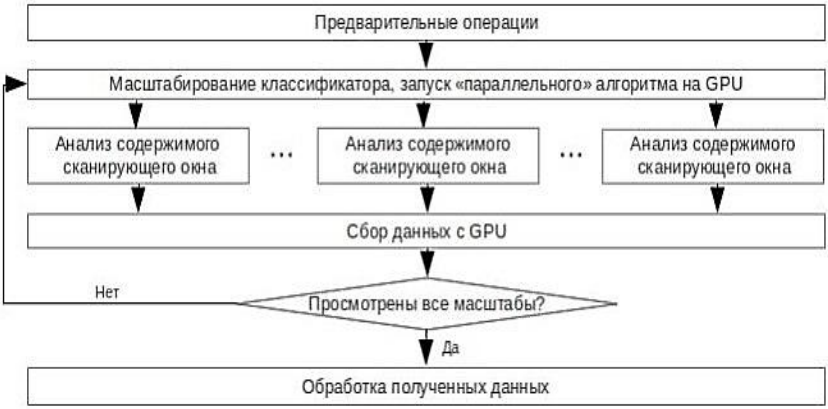

Рис. 3. Схема «параллельной» реализации алгорима Viola-Jones

Результаты замеров производительности имеющихся реализаций алгоритма показаны на рисунках 3 и 4. В качестве «последовательного» варианта использовался скрипт на языке Matlab. Система: процессор Intel Core i5-3210M 2.5 ГГц с возможностью ускорения во время выполнения до 3.1 ГГц, 6 ГБ DDR3 оперативной памяти, видеокарта – NVIDIA GeForce GT 640M с 2 ГБ памяти, установленная ОС – 64-х битная версия Windows 8.

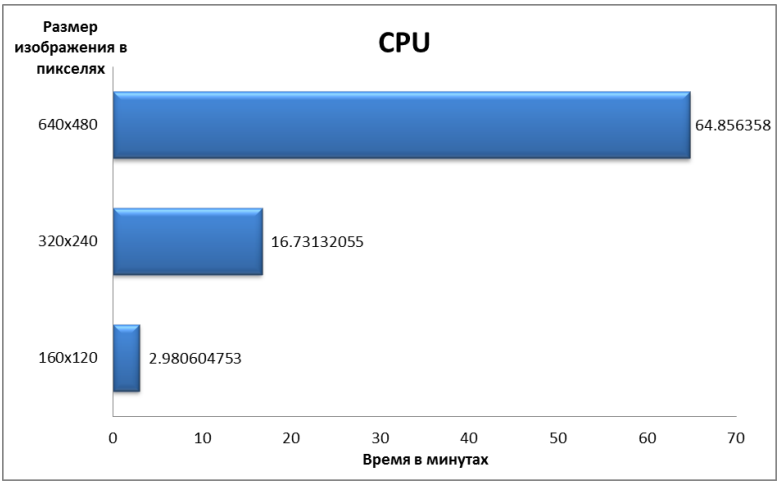

Рис. 4. Результаты замеров производительности «последовательной» версии алгоритма Viola-Jones

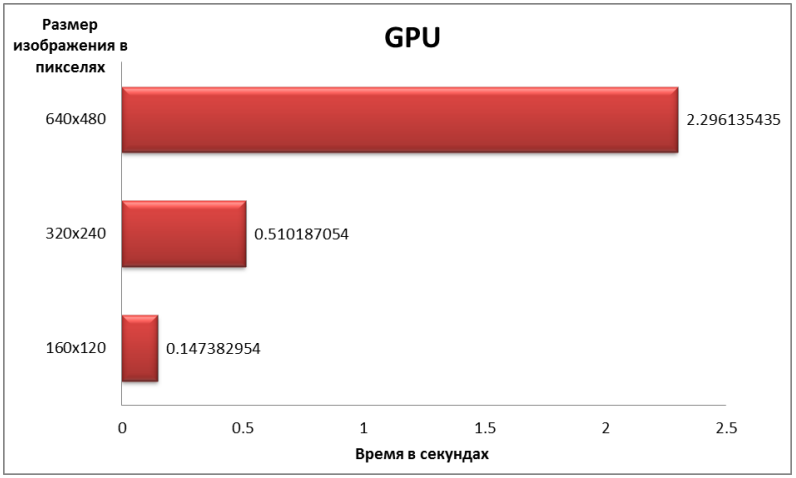

Рис. 5. Результаты замеров производительности «параллельной» версии алгоритма Viola-Jones

Результаты показывают более чем тысячекратное уменьшение времени, затрачиваемого на анализ изображения. Это позволяет

рекомендовать использование графического ускорителя при реализации метода Viola-Jones, как относительно дешевого и простого в обращении средства увеличения производительности алгоритма.

Список литературы

1. Viola P. Rapid Object Detection using a Boosted Cascade of Simple Features / Viola P., Jones M. // Proceedings of the 2001 IEEE Conference On Computer Vision And Pattern Recognition 2001. – 2001. – Vol. 1. – P. 511-518.

2. Viola P. Robust Real-Time Face Detection / Viola P., Jones M. // International Journal of Computer Vision. – Netherlands: Kluwer Academic Publishers. – 2004. – № 57(2). – P. 137-154.

3. Freund Y. A Short Introduction to Boosting / Freund Y., Schapire R. // Japan: Journal of Japanese Society for Artificial Intelligence. – 1999. – N. 14(5). – P. 771-780.

4. Harvey J. GPU Acceleration of Object Classification Algorithms Using NVIDIA CUDA : A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering / Harvey J. – Rochester, NY, 2009. – 80 p.

РАЗРАБОТКА ЭКСПЕРТНОЙ СИСТЕМЫ ПРЕЦЕДЕНТНОГО ТИПА В ОБЛАСТИ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ РЕЗЕРВНОГО КОПИРОВАНИЯ ДАННЫХ (НА ПРИМЕРЕ EMC NETWORKER SOFTWARE)

студ. И.В. Аксенова,
доц. В.В. Гаршина

Введение

EMC NetWorker software – кросс-платформенное клиент-серверное приложение, с помощью которого ООО “Atos IT Solutions and Services” осуществляет резервное копирование и восстановление серверов своих заказчиков. Для обеспечения успешной работы приложения в рамках компании функционирует служба технической поддержки, которая в соответствии с установленным уровнем обслуживания (SLA, Service Level Agreement) должна осуществлять мониторинг и устранение сбоев процессов копирования. Так как сотрудникам данной службы зачастую приходится решать одни и те же инциденты, но в параллельном режиме,

возникла необходимость разработки на основе накопленного опыта экспертной системы, которая ускорила бы процесс поиска неисправностей и их устранение.

Преимущества использования Case-Based Reasoning (CBR)

При разработке экспертной системы был выбран подход CBR. Это мощный и полезный способ решения проблем в области технической диагностики [1], который обладает следующими преимуществами:

1. CBR сокращает время поиска решения проблемы за счет использования уже имеющегося решения для схожей проблемы;
2. CBR позволяет использовать повторения ошибок. Хранение в системе информации не только о положительном, но и об отрицательном опыте, а также информации о причинах неудачи в прошлом, может предупредить большинство сбоев в будущем;
3. CBR-системы могут начать функционировать с небольшой базой прецедентов, а затем постепенно увеличивать ее;
4. CBR-системы могут работать в тех областях, которые не были полностью поняты, определены или смоделированы. Для их функционирования необходим только набор прецедентов конкретной предметной области.

Основы подхода CBR

CBR (рассуждение на основе прецедентов) - это подход, позволяющий решить новую проблему, используя или адаптируя решение уже известной задачи [2].

Прецедент обычно включает в себя:

1. Описание состояния исследуемого объекта;
2. Решение проблемы;
3. Результат применения решения [2].

Несмотря на то, что не все системы вывода на прецедентах, полностью включают этапы, приведенные ниже, в CBR можно выделить следующие компоненты:

1. *Извлечение (retrieve).* Извлечение наиболее релевантных для текущей проблемы прецедентов из библиотеки прецедентов [2];
2. *Повторное использование (reuse).* Повторное использование извлеченного прецедента для попытки решения текущей проблемы [2];

3. *Пересмотр (revise).* Пересмотр и адаптация в случае необходимости извлеченного из базы прецедентов решения в соответствии с текущей проблемой [2];

4. *Оценка применения(review).* Проверка корректности решения [3];

5. *Сохранение (retain).* Добавление текущего случая в базу прецедентов [2];

6. *Совершенствование (refine).* Индексы и весы в базе прецедентов должны быть переопределены после добавления в нее нового прецедента [3].

Архитектура экспертной системы на основе CBR

В разработанной экспертной системе представлены следующие компоненты CBR: извлечение прецедента, оценка решения, переопределение индексов и весов в базе прецедентов, добавление нового прецедента. Архитектуру программы можно представить следующим образом:

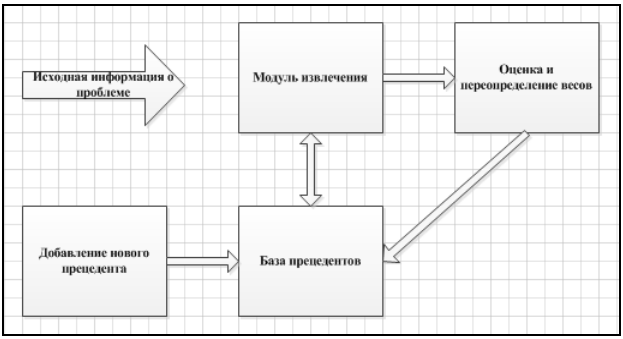

Рис.1. Архитектура экспертной системы

Проблема извлечения прецедента

В системах прецедентного типа используются разные методы извлечения прецедентов. «Сложность поиска решения и выявления различий между прецедентами в значительной степени зависит от используемых термов индексации» и представления прецедента [4]. С одной стороны, прецеденты должны обладать свойствами, связывающими их с определенными классами проблем. С другой стороны, эти свойства

должны позволить отличить определенный прецедент в рамках какого-либо класса проблем [4].

В данной работе прецеденты представлены в виде объектов, характеризующихся набором атрибутов:

$$Case_i=\{x_1,x_2,\ldots,x_h,\ldots,x_m,c^i_1,R\},$$

где $Case_i$ – исследуемый прецедент, $x_1,x_2,\ldots,x_h,\ldots,x_m$ – независимые переменные, значения которых известны (параметры ситуации, описывающей данный прецедент), c^i_1 – причина возникновения проблемы. R – это решение прецедента, зависимая переменная, значение которой определяется на основании значений $x_1,x_2,\ldots,x_h,\ldots,x_m$ и c^i_1. У каждого прецедента может быть несколько причин, по которым он возник: $C^i = \{c^i_1, c^i_2, c^i_3\}$. Каждой из причин эксперт задает начальную оценку вероятности P, которая затем динамически переопределяется.

Реализация процедуры кластерного анализа на R

Чтобы повысить эффективность работы механизма сопоставления новой проблемы с уже имеющимися прецедентами, была проверена иерархическая кластеризация. Кластерный анализ был осуществлен с помощью языка и среды программирования для статистической обработки данных R. Основа системы R – интерпретируемый язык, поддерживающий структурный, модульный и объектно-ориентированный стили программирования, а также предоставляющий программисту огромное множество встроенных статистических и графических инструментов.

В качестве меры близости между прецедентами было выбрано манхэттенское расстояние, так как для этой меры влияние отдельных выбросов меньше, чем, например, при использовании евклидова расстояния. Манхэттенское расстояние представляет собой сумму разностей по координатам и вычисляется по следующей формуле:

$$d_H(x_i,x_j)=\sum_{t=1}^{m}|x_{it}-x_{jt}|$$

Для расчета расстояния между кластерами использовался метод Уорда. Данный метод построен таким образом, чтобы оптимизировать минимальную дисперсию внутри кластеров. Эта целевая функция известна

как внутригрупповая сумма квадратов или сумма квадратов отклонений (СКО). Формула квадратов отклонений имеет вид:

$$СКО= x_j^2\text{-}1/n(\sum x_j)^2,$$

где x_j – значение признака j-го объекта. На первом шаге, когда каждый кластер состоит из одного объекта, СКО равно 0. По методу Уорда объединяются те группы или объекты, для которых СКО получает минимальное приращение. [5] Достоинством данного метода является то, что он позволяет формировать довольно компактные и хорошо разделенные кластеры.

```
setwd("D:\\")

cases1<-read.table("cases1.csv",sep=";",header=TRUE)

cases1<-cases1[-1]

cases1<-t(cases1)

d1<-dist(cases1,method="manhattan")

hc1<-hclust(d1,method="ward")

plot(hc1)
```

В результате проведенного кластерного анализа сформировалась таксономия типов прецедентов.

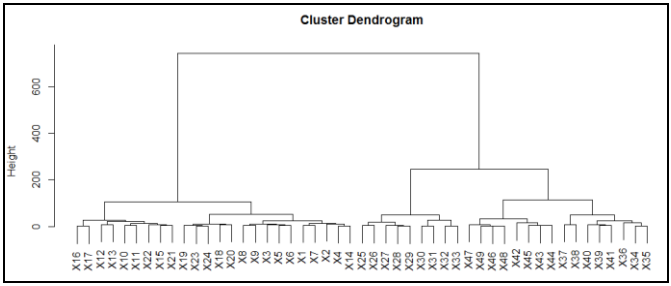

Рис.2. Дендрограмма прецедентов

Алгоритм реализации вывода в экспертной системе прецедентного типа

1. Пользователь описывает проблему набором параметров и их значений:

$Case_{new} = \{x_1, x_2, \ldots, x_h, \ldots, x_m\}$

2. Из базы прецедентов с помощью метода Манхэттена, а также с учетом значений оценок вероятности причин c^i извлекается наиболее подходящий прецедент:

$Case_i = \{x_1, x_2, \ldots, x_h, \ldots, x_m, c^i_1, R\}$

3. c^i_1 и R подставляются в $Case_{new} = \{x_1, x_2, \ldots, x_h, \ldots, x_m\}$

4. Оценка предложенного решения. Если оно оказалось удачным, то $P^i_1 = P^i_1 + 1$. Если нет, $P^i_1 = P^i_1 - 1$.

Заключение

В результате был разработан прототип экспертной системы прецедентного типа в области технической поддержки резервного копирования данных (на примере EMC NetWorker Software). Также было проведено тестирование корректности предлагаемых экспертной системой решений с привлечением экспертов данной предметной области. Оценивание проводилось по набору тестовых примеров как из предшествующей практики экспертов, так и специально подобранных ситуаций. Результаты тестирования показали высокую степень точности работы экспертной системы: для 87% тестовых примеров экспертная система предложила правильное решение.

Результаты данной работы полностью внедрены в ООО “Atos IT Solutions and Services” и успешно используются службой технической поддержки резервного копирования данных.

Список литературы

1. Richter M. Learning from HOMER, a Case-Based Help Desk Support System. / M. Richter, R. Thomas. – Berlin: IOS Press, 2003. – 97 p.

2. Aamodt A. Case-Based Reasoning: Foundational issues, methodological variations, and system approaches / A. Aamodt, E. Plaza // AI Communications. – 1994. – V.7. – P. 39-59.

3. Watson I. Case-Based Reasoning: A Review. / I. Watson, F. Marir. // Режим доступа: http://www.ai-cbr.org/classroom/cbr-review.html. - 2006.

4. Джексон П. Введение в экспертные системы: пер. с англ.: Уч. пос. / П. Джексон. – М.: Издательский дом «Вильямс», 2001. – 624 с.

5. Aldenderfer M.S. Cluster Analysis / M.S. Aldenderfer, R.K. Blashfield. - London: Sage press, 1986. – 211 p.

НЕЙРОСЕТЕВЫЕ МОДЕЛИ И АЛГОРИТМЫ РАСПОЗНАВАНИЯ ЗАБОЛЕВАНИЙ В СТОМАТОЛОГИИ НА ОСНОВЕ АНАЛИЗА СПЕКТРАЛЬНЫХ ХАРАКТЕРИСТИК РАССЕЯННОГО ЛАЗЕРНОГО ИЗЛУЧЕНИЯ

студ. Л.В. Баркалова,
доц. Д.А. Минаков,
проф. А.А. Сирота

В течение многих лет исследователи стремились улучшать существующие традиционные методы диагностики заболеваний твердых тканей зубов, такие как клиническое обследование или рентгеноскопия, которые чаще всего используются в клинической практике [1, 2]. При этом основная цель при проведении таких исследований состоит в идентификации патологий еще на ранних стадиях при соблюдении высокой точности диагностики. К настоящему времени созданы новые методы диагностики, которые являются достаточно эффективными, но пока не нашли широкого применения в клинике. Расширенный обзор новых технологий диагностики можно найти в литературе [1, 2].

Одним из наиболее перспективных и неинвазивных методов диагностических процедур является метод лазерно-индуцированной флуоресценции (ЛИФ), который уже достаточно давно и успешно используется, например, для определения различий между нормальными и атеросклеротичными тканями, а также для регистрации рака.

Данный метод основан на измерении и интерпретации спектров флуоресценции, индуцированной в исследуемом объекте при его освещении монохроматическим лазерным излучением. Измеряемый спектр представляет собой совокупность флуоресцентных откликов индивидуальных компонентов, присутствующих в исследуемом образце. Поскольку спектральный состав флуоресцентного отклика связан с молекулярной структурой исследуемых объектов, анализ спектров флуоресценции во многих случаях позволяет идентифицировать вещество

и определить его концентрацию в объекте. Относительно давно метод ЛИФ используется и в стоматологии.

Цель данной работы заключается в исследовании возможности диагностики кариеса зубов человека с помощью нейросетевых моделей и алгоритмов, а также создании, обучении и совершенствовании нейросетевых алгоритмов, анализирующих спектральные данные ранней диагностики кариеса на основе метода ЛИФ.

Для этого были использованы результаты исследования флуоресценции здоровых участков твердых тканей зубов и участков, пораженных кариесом на различных стадиях его развития. Регистрация спектров флуоресценции была осуществлена на экспериментальной установке, позволяющей регистрировать слабые световые потоки в режиме реального времени.

Исследования проводились на удаленных зубах людей. Все зубы имели кариозное поражение, согласно предварительным клиническим и рентгенологическим исследованиям. Стадии кариозного процесса определяли в соответствии с топографической классификацией: поверхностный кариес, средний кариес, глубокий кариес. Спектр флуоресценции измеряли с использованием экспериментальной установки, созданной на базе волоконно-оптического спектрометра USB4000-VIS-NIR (350 - 1000 нм) фирмы Ocean Optics. Данная установка предназначена для регистрации флуоресценции локальных областей объектов спектральных медицинских исследований и может быть использована для спектральной диагностики внутренних и поверхностных областей различных биологических сред. Схема экспериментальной установки приведена на рис.1.

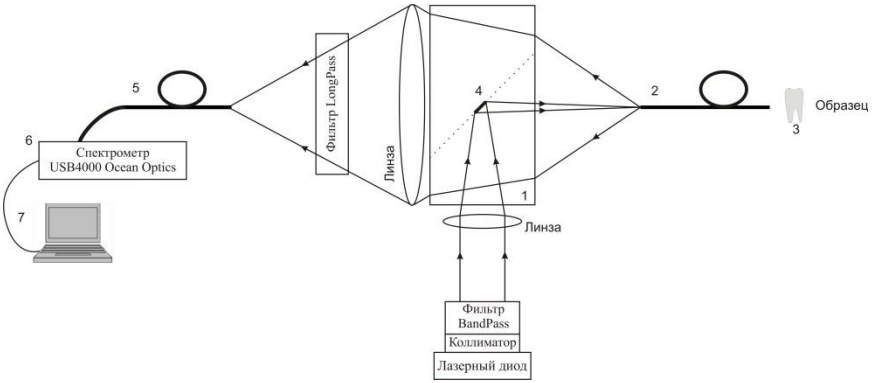

Рис.1. Схема экспериментальной установки для изучения спектров флуоресценции: 1 – делительная призма; 2, 5 – оптический волновод; 3 – образец; 4 – зеркальное покрытие; 6 – спектрометр

На рис.2 представлены усредненные спектры флуоресценции от интактной твердой ткани зубов человека и от тканей, пораженных кариесом различных стадий его развития, а именно, поверхностной, средней и глубокой. Источник возбуждения - лазерный диод, излучающий на длине волны 405 нм. Наблюдается непосредственная связь между стадией кариозного поражения и формой спектра флуоресценции [3].

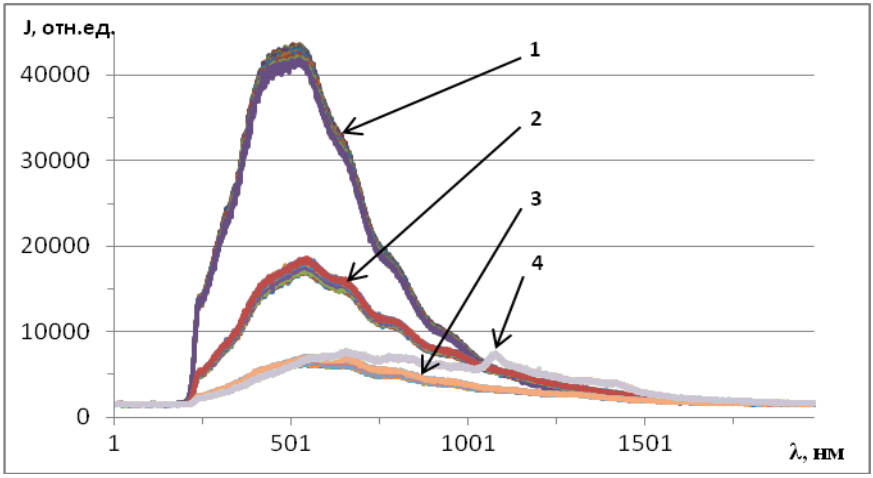

Рис.2. Спектры флуоресценции зубов человека: 1 - интактная твердая ткань зуба; 2 - ткань, пораженная поверхностным кариесом; 3 - средним кариесом; 4 - глубоким кариесом

Как видно, метод лазерно-индуцированной флуоресценции достаточно надежен, но возникает проблема быстрого анализа полученных данных и наглядности результата этого анализа. Так как объем данных флуоресцентной спектроскопии очень велик, необходимо создать такую программу, которая могла бы обрабатывать большое количество информации за минимальное время с высокой точностью. Поэтому для решения этой задачи целесообразно использовать модель искусственной нейронной сети (ИНС). Для работы с ИНС использовалась среда MATLAB.

Полученные спектры флуоресценции анализировалась с помощью двухслойной нейронной сети с 20 нейронами на внутреннем слое, логарифмической сигмоидной и линейной функциями активации и функцией обучения сети «trainrp», которая модифицирует веса и смещения в соответствии с алгоритмом упругого обратного распространения [4]. Размер обучающей выборки – 800 спектров (по 200 спектров для каждой стадии кариозного поражения), количество вариантов выхода сети – 4 (рис.3).

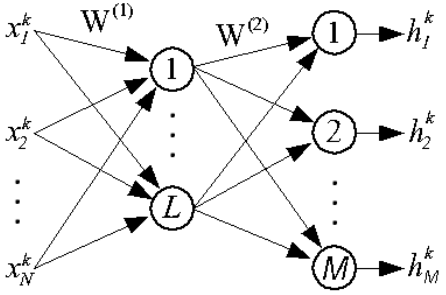

Рис.3. Схема искусственной нейронной сети для классификации спектров флуоресценции: количество входов сети N = 800, количество нейронов на внутреннем слое L = 20, количество вариантов выхода сети M = 4

Пространственное отображение спектров флуоресценции по классам было получено с помощью метода выделения главных компонент. Из рис.4 видно, насколько трудно визуально разделить стадии кариозного процесса на необработанных данных.

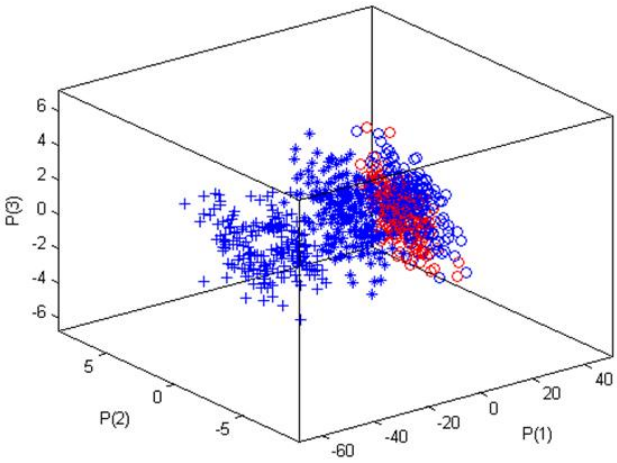

Рис.4. Отображение данных методом выделения главных компонент

Поэтому перед началом обучения сети проводилась предварительная нормировка данных по следующей формуле:

$$\tilde{x}_i = \frac{x_i}{\sum_{i=1}^{N} x_i} \quad , \qquad (1)$$

где значение каждого элемента отдельно взятого спектра флуоресценции делится на сумму значений всех элементов этого спектра. Таким образом, все спектры по интенсивности флуоресценции будут приведены к одному диапазону, и далее будет анализироваться только форма спектров.

Обучение и тестирование нейронной сети проводилось в 3 экспериментах. В первом эксперименте тестирование сети производится на обучающих данных. Возможно внесение зашумления в тестируемые данные путем добавления гауссовских возмущений. В результате тестирования сеть определяет стадию кариозного поражения зубной ткани с вероятностью, равной 1.

Во втором и третьем экспериментах тестирование сети производится на данных, не используемых при обучении с возможностью дополнительного зашумления; в третьем эксперименте предварительно проводится искуственное размножение обучающих данных. В результате тестировния сети получен результат в промежутке от 0,93 до 0,96 в зависимости от степени зашумленности исходных данных. Например, при

внесении зашумления в тестируемые данные путем добавления гауссовских возмущений с СКО sigma = 0,03 матрицы ошибок выглядят следующим образом:

```
matros2 =
    0.9900         0    0.0100         0
    0.0100    0.8850    0.0800    0.0250
    0.0100    0.0050    0.9700    0.0150
    0.0100    0.0050    0.0550    0.9300

matros3 =
    0.9950         0    0.0050         0
    0.0100    0.9000    0.0700    0.0200
    0.0050         0    0.9800    0.0150
         0         0    0.0600    0.9400

probmean2 =
    0.9437

probmean3 =
    0.9537
```

Анализ полученных результатов позволяет сделать вывод о том, что рассмотренный алгоритм позволяет потенциально обеспечить достаточно высокую достоверность распознавания стадии кариозного поражения зубной ткани. Полученные средние минимальные вероятности ошибок распознавания составляют порядка 0.05%.

Достоинством нейросетевого алгоритма является его большая устойчивость к возможным неспецифичным изменениям характера входных данных спектров флуоресценции, связанных с естественной изменчивостью объектов в рамках одной или нескольких выборок. При использовании искусственных нейронных сетей всегда существует возможность осуществления процедуры их дообучения с использованием новых данных или обучения новой сети без принципиального изменения схемы работы самого алгоритма распознавания. Также следует отметить возможность повышения качества распознавания объектов по измеренным спектрам за счет увеличения размерности представления входных данных.

Список литературы

1. Pretty I.A. Caries detection and diagnosis: novel technologies. – J Dent Res, 2006; 34:727-739.
2. Higham S.M. Application biophysical technologies in dental research. /S.M. Higham, N. Pender, E.J. Jong, P.W. Smith/ – J Appl Phys, 2009; 105:102-048.
3. Сарычева И.Н. Ранняя диагностика зубного кариеса методом лазерно-индуцированной флуоресценции /И.Н. Сарычева, О.О. Янушевич, Д.А. Минаков, В.А. Шульгин, В.М. Кашкаров/ – Воронежский Государственный университет, Воронеж, 2012.
4. Заенцев И.В. Нейронные сети: основные модели. Учебное пособие для студентов 5 курса магистратуры к. электроники физического факультета Воронежского Государственного университета – Воронежский Государственный университет, Воронеж, 1999.
5. Ануфриев И.Е. MATLAB 7. /И.Е. Ануфриев, А.Б. Смирнов, Е.Н. Смирнова/ – СПб: БХВ-Петербург, 2005; 78-80, 213-234.

WEB-ПРИЛОЖЕНИЕ «СЕРВЕР ОТЧЕТОВ»

студ. Г. Н. Вознесенская,
ст. преп. Д. И. Соломатин

Построение отчетов являются неотъемлемой частью любого бизнес-приложения. По своей природе отчеты могут быть условно разделены на два класса, которые можно назвать печатными формами и аналитическими отчетами. Первые предназначены для создания печатных документов, в которых помимо данных важна также форма представления этих данных на листе бумаги (форма, например, может быть утверждена различными нормативными положениями).

В отличие от печатных форм в аналитических отчетах, прежде всего, важны собственно данные, а требования к форме их представления минимальны. Как привило, данные в таких отчетах представлены в виде простых таблиц, в некоторых случаях требуется наглядное графическое представление табличных данных в виде графика или диаграммы. Однако, учитывая, что такие отчеты предназначены, прежде всего, для анализа информации, в них часто требуется некоторая интерактивность, а именно возможность параметризации выбираемых данных, а также их сортировки по различным параметрам.

В приложениях с базами данных суть аналитического отчета, как правило, может быть выражена в виде параметризованного sql-запроса (или нескольких запросов). Однако разработчику приложения приходится дополнительно писать однотипный код для создания формы для ввода параметров отчета, для вывода тела отчета и т.п. Идея данной работы состоит в создании инструмента, позволяющего упростить создание аналитических отчетов согласно формуле «создание отчета = написание соответствующего sql-запроса».

Для решения данной задачи разработано web-приложение «сервер отчетов». Помимо собственно построения отчетов сервер обеспечивает разграничения прав доступа пользователей к отчетам на сервере.

Отчеты для данного приложения описываются в виде xml-файлов. Помимо собственно sql-запроса для построения отчета в данном xml-файле описываются параметры отчета. Общая структура xml-файла описания отчета представлена ниже:

```
<?xml version="1.0"?>

<report>

  <name>название отчета</name>

  <type>тип отчета (grid/bar/column/pie/pie donut/

                    line smoothed/line step)</type>

  <!-- описание параметров соединения с БД -->

  <connection>

    <dsn>строка связи (для PDO)</dsn>

    <username>имя пользователя</username>

    <password>пароль пользователя</password>

  </connection>

  <!-- описание параметров отчета -->

  <params>

    <param>

      <name>название параметра</name>
```

```
        <type>тип параметра (search/input/number/date/
                              select/radio)</type>
        <sql>sql-запрос для выборки возможных значений
             параметра (только для некоторых типов)</sql>
        <depended>зависимые параметры</depended>
        <condition>условие зависимости</condition>
      </param>
      ...
    </params>
    <!-- sql-запросы для выборки данных для отчета -->
    <queries>
      <query>
        <name>название</name>
        <sql>собственно запрос</sql>
      </query>
      ...
    </queries>
  </report>
```

Создание отчета происходит следующим образом. Сначала пишется параметризованный sql-запрос, который составляет суть отчета. Написание и отладка данного sql-запроса может происходить в любой среде разработки для работы с требуемой СУБД. Затем составленный запрос вставляется в шаблон отчета, а также описываются параметры отчета. Получившийся xml-файл загружается на сервер отчетов. Таким образом, разработка отчета, по сути, сводится к написанию sql-запроса (запросов), никаких специальных инструментов (дизайнеров отчетов и т.п.) для создания отчетов и их правки не требуется.

Реализация приложения выполнена на языке PHP. Для взаимодействия с базами данных использовался универсальный интерфейс доступа PDO,

таким образом, поддерживается работа со всеми распространенными СУБД. Клиентский интерфейс выполнен с использованием javascript-библиотек jQuery [1] и jQuery EasyUI. Все данные между клиентом и сервером передаются по протоколу HTTP в формате JSON [2].

Рассмотрим работу приложения более подробно. При выборе пользователем какого-либо отчета из списка ответов, представленных на сервере, отображается форма, позволяющая указать параметры отчета в соответствии с его описанием, см. рис. 1.

Рис. 1. Форма ввода параметров отчета

После нажатия кнопки «Execute» происходит отправка параметров на сервер и получение данных для отчета. В случае табличного отчета данные отображаются в виде таблицы, см. рис. 2. Также появляется панель для выбора количества показанных записей и переключения страниц.

Detailed information

	first_name	last_name	department_name	city
1	Adam	Fripp	Shipping	South San Francisco
2	Alexander	Hunold	IT	Southlake
3	Bruce	Ernst	IT	Southlake
4	David	Austin	IT	Southlake
5	Diana	Lorentz	IT	Southlake
6	Matthew	Weiss	Shipping	South San Francisco
7	Michael	Hartstein	Marketing	Toronto
8	Pat	Fay	Marketing	Toronto
9	Payam	Kaufling	Shipping	South San Francisco
10	Valli	Pataballa	IT	Southlake

10 | Page 1 of 7 | Displaying 1 to 10 of 70 items

Рис. 2. Вид табличного отчета

В случае графического отчета пользователь увидит представление выбранных данных в виде диаграммы, указанной в описании отчета, см. Рис 3.

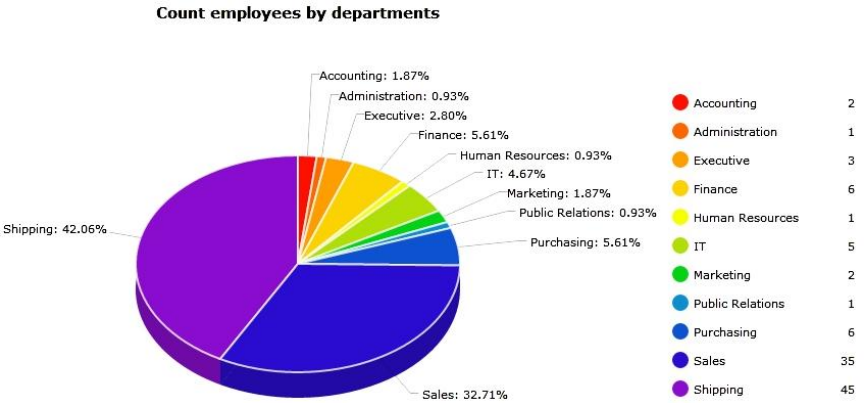

Рис. 3. Вид графического отчета (тип pie)

В результате проделанной работы было создано приложение, которое позволяет строить различные аналитические отчеты. Основным преимуществом данного приложения является простота создания отчетов.

В дальнейшем планируется дорабатывать данное приложение с целью придания ему еще больших возможностей и гибкости.

Список литературы

1. Bibeault Bear. jQuery in Action, Second Edition / Bear Bibeault, Yehuda Katz – Manning Publications, 2010. – 475 p.
2. Cristian Darie. AJAX and PHP: Building Responsive Web Applications / Cristian Darie, Bogdan Brinzarea, Filip Chereches-Tosa, Mibai Bucica – Packt, 2007. – 334 p.

АНАЛИЗ АЛГОРИТМА УСИЛЕНИЯ АМПЛИТУДЫ ВЕРОЯТНОСТИ

студ. Н.В. Воронцова,
проф. С.А. Запрягаев

Известный квантовый алгоритм Гровера решает задачу поиска заданного элемента в неупорядоченной базе данных с использованием свойства усиления амплитуды вероятности квантового состояния при выполнении определенной процедуры с квантовым регистром. Алгоритм усиления амплитуды вероятности, заключается в том, чтобы найти помеченный элемент из базы данных, и основан на последовательности действий с амплитудами всех элементов базы данных.

Таким образом, целью данной работы является проведение анализа алгоритма усиления вероятности помеченного элемента базы данных, а также создание программного комплекса, позволяющего провести этот анализ.

Пусть есть неупорядоченная база данных, содержащая N элементов $x_1, x_2, x_3, ...x_N$. Этими элементами могут быть любые сущности: номера телефонов, фотографии идентифицирующие людей, детали какого-то производства и так далее. Если требуется найти в этой базе конкретный элемент, который обозначим через z, то стандартная процедура отыскания этого элемента может состоять в случайном переборе элементов базы x_i и "сличении" их с образцом для поиска z. Когда окажется, что $x_i = z$ можно утверждать, что искомый элемент найден. Очевидно, что если размер базы N, то может потребоваться $O(N)$ испытаний для отыскания конкретного элемента.

Обозначим амплитуду вероятности извлечения x_i-го элемента через a_i и допустим, что вероятности $p_i = |a_i|^2$ извлечения из базы всех элементов одинаковы $a_i = 1/\sqrt{N}$ Выполним циклически операции определенные следующими тремя шагами:

1 шаг. Вычисление среднего значения амплитуды $< a >$ для всех элементов базы данных: $< a >= \frac{1}{N}\sum_{i=1}^{N} a_i$;

2 шаг. Преобразование амплитуд a_i в соответствии с выражением:

$a_i = 2 < a > -a_i, i \in 1,2,...,N$;

3 шаг. Изменение знака амплитуды искомого элемента базы данных на противоположный: $a_p = -a_p$.

Изменение амплитуды вероятности выделенного элемента базы данных до практически достоверного события и является проявлением усиления амплитуды вероятности нахождения данного элемента в неупорядоченной базе данных. Рассмотрим реализацию алгоритма на простом примере. Зададим количество элементов $N = 32$, промаркируем 5 элемент и зададим выполнение 15 итераций.

На рис.1 представлены результаты работы программы, выводящей график зависимости амплитуды вероятности маркированного элемента от числа итераций, а также таблица значений всех амплитуд.

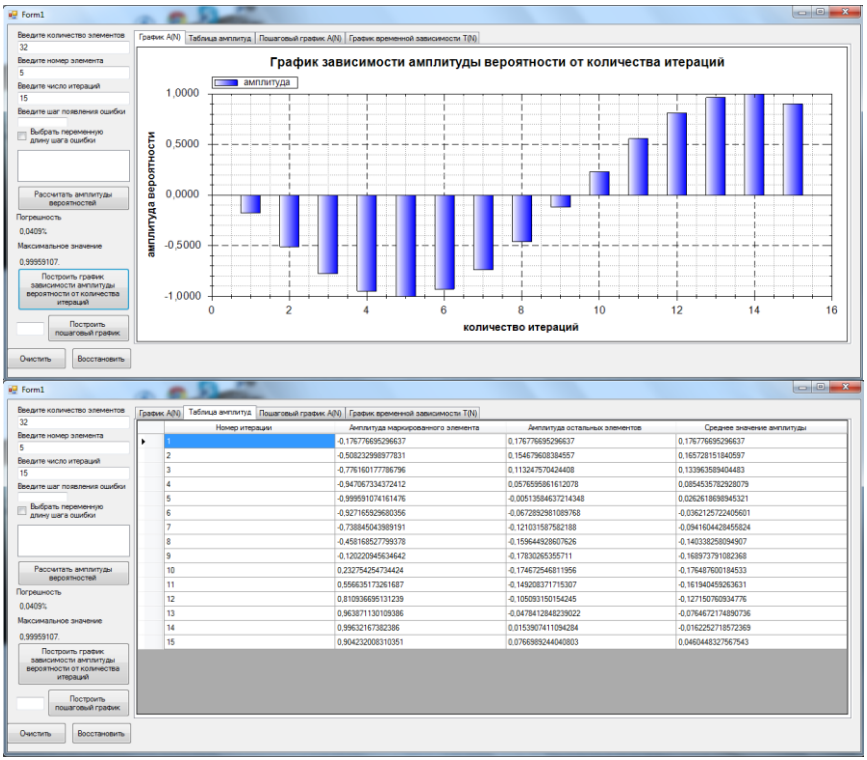

Рис.1. График зависимости амплитуды вероятности от количества итераций и таблица значений всех амплитуд

Из рис.1 видна динамика изменения амплитуды вероятности маркированного элемента после каждой итерации. Следует обратить внимание, что на пятой итерации амплитуда искомого состояния достигает максимума равного = 0,999, а затем начинает колебаться. То есть на пятой итерации, что по порядку величины совпадает с $\approx\sqrt{N}$, вероятность того, что из базы данных будет извлечен искомый элемент, практически равна достоверному событию.

В настоящей работе были проанализированы два метода нахождения маркированного элемента из базы данных. Первый метод - усиление амплитуды вероятности маркированного элемента в соответствии с изложенным выше алгоритмом преобразования амплитуд. Второй метод - случайная выборка элемента. Следует заметить, что при реализации случайной выборки элемента необходимо учитывать возможность повторения чисел в стандартном классе System.Radom, что может привести к случаю, когда, перебрав N элементов, искомый маркированный элемент так и не будет найден. Поэтому необходимо ввести условия, которые будут отсеивать все повторяющиеся ранее значения случайной выборки, что позволит гарантированно найти маркированный элемент в базе данных. После создания случайного числа и сравнения с номером маркированного элемента, его необходимо запоминать, чтобы далее не выполнять проверку повторно. Для этого формируется динамический список, в который попадает случайное число на каждом шаге. При этом каждый раз проводится проверка поиска вновь созданного случайного числа в динамическом списке, если случайное число уже встречалось, то выбирается новое случайное число и проверка проводится вновь.

Проведены статистические испытания, в результате которых было установлено, что если количество элементов сравнительно небольшое, то эффективность этих алгоритмов примерно одинакова. Так как в некоторых случаях быстрее достигается поиск по методу усиления амплитуды, что требует порядка $\approx\sqrt{N}$ шагов. Однако в некоторых случаях метод случайной выборки быстрее попадает на искомый элемент. Однако чем большее количество элементов рассматривается, тем заметнее становится различие между двумя способами.

В результате при количестве элементов $N>100$, метод усиления амплитуды вероятности становится гораздо более эффективным, чем метод случайной выборки элемента. Этот факт продемонстрирован на рис.2 в виде двух графиков. Нижний график построен по точкам, полученным с помощью алгоритма усиления амплитуды вероятности, а верхний – с помощью метода случайной выборки.

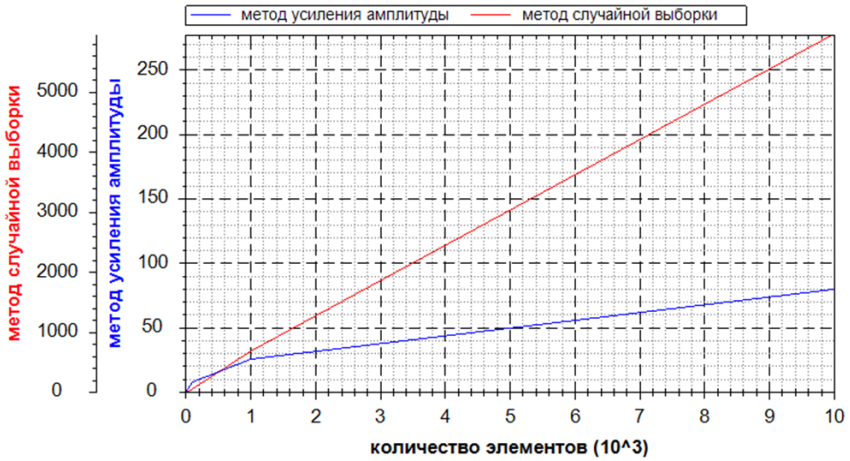

Рис.2. График зависимости времени необходимого для поиска маркированного элемента из базы данных, содержащей N образцов

Кроме выполнения анализа эффективности алгоритмов в работе проведена оценка влияния ошибки на работу алгоритма усиления амплитуды вероятности. Ошибка определяется тем, что через заданное количество шагов в алгоритме возникает сбой. Был сымитирован сбой программы, при котором третий шаг алгоритма усиления амплитуды вероятности не выполняется, то есть знак амплитуды вероятности маркированного элемента не меняется. Результат анализа демонстрирует, что если шаг ошибки оказывается большим, чем $\approx\sqrt{N}$, то на амплитуде вероятности маркированного элемента это существенно не отражается, поскольку она успевает достигнуть максимума. Поэтому шаг ошибки, выбирается меньше чем $\approx\sqrt{N}$. Проанализировав и проведя статистические измерения, представленные в табл.1, можно сделать следующий вывод.

Вывод: чем меньший выбирается шаг ошибки, тем большую погрешность будет иметь алгоритм усиления амплитуды. Чем ближе шаг ошибки находится к значению $\approx\sqrt{N}$, тем погрешность алгоритма будет меньше. Это наглядно продемонстрировано в табл.1, где также представлены статистические измерения интересующих величин.

Таблица 1 – Статистическая таблица максимальных значений амплитуд от длины ошибки

Шаг ошибки	Количество элементов		
	10^3	10^4	10^5
$\sqrt{N} + \sqrt{N}/2$	0,9997	0,9999	0,9999
$\sqrt{N}$	0,9997	0,9999	0,9999
$\sqrt{N}/2$	0,7939	0,8360	0,8393
$\sqrt{N}/3$	0,6164	0,6051	0,6138
$\sqrt{N}/4$	0,4568	0,4706	0,4763
$\sqrt{N}/5$	0,3409	0,3810	0,3850
$\sqrt{N}/6$	0,2808	0,3204	0,3199
$\sqrt{N}/7$	0,2195	0,2667	0,2777
$\sqrt{N}/8$	0,2195	0,2279	0,2410
$\sqrt{N}/9$	0,1574	0,2084	0,2164
$\sqrt{N}/10$	0,1574	0,1886	0,1917

Но в реальных системах крайне редко ошибки и сбои происходят через определенное число шагов и носят систематический характер, гораздо чаще появление сбоя происходит непредсказуемо. С целью исследования этого варианта был рассмотрен случай, когда шаг ошибки носит случайный характер. Проведены статистические испытания, в результате которых установлено, что если появление ошибки носит случайный характер, то амплитуда вероятности в большинстве случаев достигает максимального значения ≈1, а погрешность алгоритма останется незначительной. Можно заметить, что существуют такие испытания, при которых случайный характер появления ошибки существенно повлиял на алгоритм усиления, но в подавляющем большинстве погрешность была меньше 1%.

Вывод: алгоритм усиления вероятности амплитуды более устойчив к появлению ошибки, которая носит случайный характер, нежели к систематической ошибке, частота появления которой значительно меньше чем $\approx \sqrt{N}$.

Следует отметить, что раньше при проведении статистических испытаний время, затраченное на нахождение элемента алгоритмом

усиления амплитуды вероятности, было всегда одинаковым при одинаковом числе элементов в базе данных. Теперь время существенно различается на протяжении всех испытаний. Это связано с влиянием ошибок, которые оказывают действие на общее поведение амплитуд вероятности маркированного элемента, которое можно увидеть на рис.3.

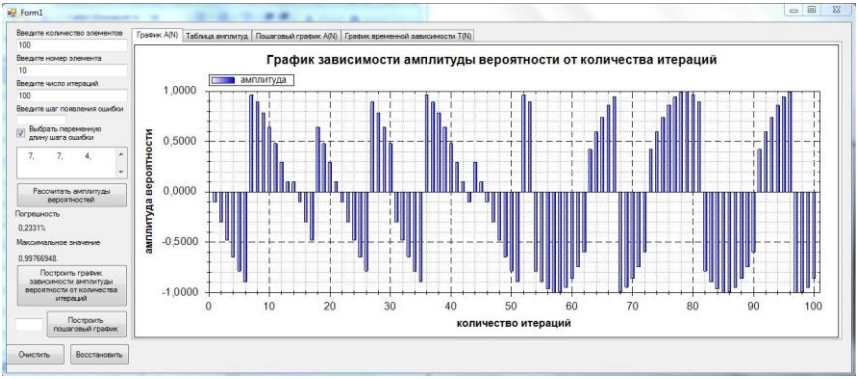

Рис.3. График зависимости амплитуды вероятности от количества итераций с варьируемым шагом ошибки

Разработка системы велась на языке C# в Visual Studio 2010 под .Net Framework 4.0 с использованием подключаемой библиотеки zedgraph.dll.

Результатом представленной работы является система, удовлетворяющая ряду требований, выполнение которых позволяет провести анализ процесса усиления амплитуды.

Список литературы

1. Китаев А. Классические и квантовые вычисления /А. Китаев, А. Шень, М.Вялый // М.МЦНМО, 1999. – 192с.
2. Кайе Ф. Введение в квантовые вычисления / Ф. Кайе, Р. Лафламм, М. Моска. – М: Институт компьютерных исследований. 2009. – 338с.
3. Нильсен М. Квантовые вычисления и квантовая информация / М. Нильсен, И. Чанг. – М.: Мир, 2006. – 824с.
4. Валиев К.А. Квантовые компьютеры: надежды и реальность / К.А.Валиев, А.А. Кокин. – Ижевск: НИС "Регулярная и хаотическая динамика", 2001. – 352 с.

ОПЫТ ИСПОЛЬЗОВАНИЯ API-БИБЛИОТЕК ДЛЯ РАБОТЫ С ЭЛЕКТРОЭНЦЕФАЛОГРАФАМИ СЕМЕЙСТВА НЕЙРОН-СПЕКТР

студ. М. Н. Гончаров,
доц. Я. А. Туровский

В настоящее время существуют медицинские приборы для регистрации практически любых параметров функциональных и морфологических особенностей человека. Такое оборудование всё чаще производят с возможностью подключения к компьютеру. Обусловлено это несколькими причинами. Во-первых, снижение конечной стоимости продукта в связи с использованием более дешёвых компонентов или исключения некоторых компонентов вообще, ведь обработку показаний и их отображение теперь выполняет компьютер, к которому подключен прибор. Вторая причина состоит в том, что у такого оборудования увеличивается мобильность. Связано это всё с тем же исключением некоторых компонентов, например, монитора. Третья причина – это ограниченные возможности автономного оборудования, а также сложность добавления дополнительных функций. Связано это с тем, что такие приборы лишены простого способа обновления программного обеспечения. Обычный компьютер же предоставляет большую гибкость в этом вопросе. Однако даже в этом случае программное обеспечение не всегда удовлетворяет потребности клинической и фундаментальной медицины.

В ряде случаев требуются специфические расчёты, основанные на параметрах, зарегистрированных аппаратной частью медицинского прибора. Подобное необходимо при особой диагностике заболеваний, однако чаще всего используется в исследованиях и экспериментах. К сожалению, производитель оригинальных программ не может предоставить весь требуемый исследователями функционал в своём программном обеспечении. Компания выпускает программное обеспечение с базовым функционалом, удовлетворяющим значительную часть медицинского сообщества, однако для полноценных клинических фундаментальных исследований подчас необходимы значительные улучшения, связанный со спецификой работы той или иной клиники, отделения, а иногда и конкретного специалиста. Для того чтобы следовать пожеланиям заказчиков, фирмам-производителям медицинской техники требуется огромный штат программистов, что, в свою очередь, не каждая фирма может себе позволить. С другой стороны индивидуальные заказы на совершенствование программного обеспечения к уже существующим приборам не только достаточно дороги, но и выполняются в течение

достаточно длительных промежутков времени, что накладывает серьёзные ограничения на характер клинических и фундаментальных исследований, снижая их информативность и подчас экономическую эффективность. Этим обусловлен рост потребности программистов в медицинской сфере.

В настоящее время ООО «Нейрософт» и ООО «Петр Телегин» бесплатно прилагают к своим приборам API-библиотеки. Как правило, эти библиотеки обладают минимумом необходимых функций и процедур для работы с прибором, обычно их возможности ограничиваются получением необработанных показаний и изменением режимов работы прибора. Тем не менее, данный минимум открывает большие возможности для исследований и экспериментов, в первую очередь за счёт того, что непосредственно с приборов получаются исходные данные, не обработанные фильтрами, которое работают подчас по закрытым для пользователей формулам.

В работе были использованы API-библиотеки для работы с электроэнцефалографом Нейрон-Спектр-4/ВП, предоставленные компанией Нейрософт. API состоит из трёх файлов: *EEG4DLL.dll*, *NS4M.dll* и NsEzUSB.dll. В библиотеке *EEG4DLL.dll* хранятся процедуры регистрации данных, поступающих с прибора, а также управления его режимами работы. Две другие библиотеки в большей степени используются внутри первой библиотеки и содержат методы более низкого уровня, например, функции управления интерфейсом USB. В связи с этим, файлы *NS4M.dll* и *NsEzUSB.dll* представляют малый интерес и в данных исследованиях напрямую не применялись.

Библиотека *EEG4DLL.dll* содержит богатый набор функций и процедур, позволяющий без лишней сложности получить доступ к необходимым данным, регистрируемым прибором. Эти методы можно разбить на несколько групп: управление состоянием прибора, управление режимами работы и управления стимуляторами. К первой группе относятся такие функции, как включение и выключение прибора, управление светодиодами на лицевой панели, получение информации о приборе и калибровка. Все они позволяют довольно просто контролировать общее состояние энцефалографа. Например, для того, чтобы включить прибор, достаточно подключить его к порту USB и выполнить функцию *SwitchOn* с параметром, указывающим на версию прибора, в рассматриваемом случае он равен 6. В случае удачи функция вернёт информацию об интерфейсе, к которому было осуществлено подключение. Если имеется несколько приборов, то можно использовать функцию *SwitchOnPid,* которая позволяет включить оборудование по идентификатору. Процедура управления светодиодами – *LightOnLed* также проста, в неё достаточно

передать массив чисел, каждый элемент означает цвет отдельного индикатора.

Вторая группа - управление режимами передачи данных представляет больший интерес. В её состав входят режим передачи электроэнцефалограммы (ЭЭГ), режим передачи вызванных потенциалов (ВП), режим измерения импеданса и поляризации. Все эти процедуры основаны на функциях обратного вызова. Это означает, что одним из её параметров является функция с определённым набором переменных. После инициализации процедуры запуска режима библиотека сама вызывает указанную функцию и передаёт в неё данные, зарегистрированные прибором. Для примера рассмотрим процедуру *SetTransmitEEG,* которая включает режим передачи ЭЭГ. В качестве параметра ей необходимо передать функцию вида:

func_user_eeg(type_buffer_eeg buffer, long length, int time_on_bus);

Здесь buffer – указатель на двумерный массив long[COUNT][length] [1]. Массив содержит отсчеты АЦП, которые были необходимы для моего исследования. В последствие они записывались и обрабатывались для получения необходимой информации. COUNT – количество передаваемых каналов ЭЭГ [1]. length – количество отчетов одного канала. *time_on_bus* – счетчик шины USB, значение которого указывает на время отчета первого элемента в массиве [1].

Преимущество такого вида передачи очевидно – это простота получения данных с прибора. Минусом же можно назвать необходимость знать точный вид функции обратного вызова, что без хорошей документации узнать крайне трудно.

Третья группа функций – управление стимуляторами, позволяет большое количество вариантов стимуляций при помощи уже имеющихся процедур, что способствует ускорению разработки программы. Например, в состав библиотеки входят такие функции, как установление периода стимуляции, количества стимулов, амплитуды токового и зрительного стимула, длительности токового стимула, поля паттерна, параметров слухового частотного стимула, параметров слухового импульсного стимула, амплитуды слухового стимула и прочие. В целом они мало отличаются от процедур управления состояниям прибора и довольно просты в применении.

Все API можно охарактеризовать двумя параметрами: набор функций и процедур и качественная документация. Даже учитывая, что вышеописанные библиотеки содержат в себе минимальный функционал, они вполне удовлетворяют потребностям разработчиков. Наличие же подробной документации способствует быстрому внедрению их в проект.

В заключение можно с уверенностью сказать, что библиотеки компании «Нейрософт» позволяют в полной мере использовать весь потенциал оборудования.

Список литературы

1. Официальная документация по API-библиотекам электроэнцефалографа Нейрон-Спектр-4/ВП.

АВТОМАТИЗИРОВАННОЕ РАБОЧЕЕ МЕСТО ОПЕРАТОРА ВЫПЛАВКИ СТАЛИ

студ. К.Б. Григорчук,
проф. М.Г. Матвеев

Важную роль в развитии и оптимизации современного металлургического производства играют системы управления, построенные на математических и компьютерных моделях. Производство современной высококачественной стали требует применение интеллектуальных наукоемких технологий. Это обусловлено быстро возрастающим уровнем требований к комплексу специальных заказных свойств металлопродукции. В частности, таким свойством является прокаливаемость стали, представляющая собой распределение твердости стальной отливки от поверхности к центру.

Точно задать эту твердость невозможно, поэтому на практике задается некоторый коридор, в котором должна находиться получаемая прокаливаемость (рис. 1).

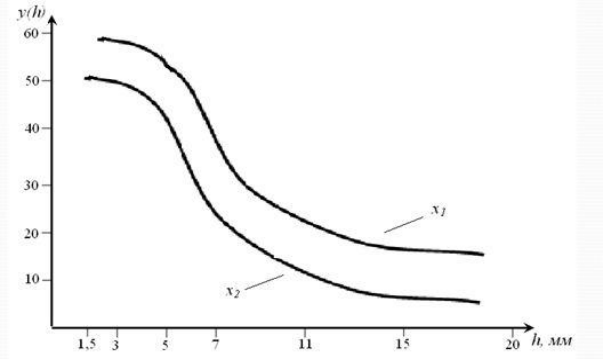

Рис. 1. График изменения твердости стали по ее глубине

Обеспечение необходимой твердости достигается, в значительной степени, за счет заданного начального химического состава, используя сложные системы микролегирующих добавок в металле.

Проблема заключается в выборе такого начального химического состава, который обеспечит попадание в заданный коридор. Раньше этот выбор осуществлялся операторами плавки на основании их опыта и знании представленных различного рода номограммами. Несколько лет назад немецкая компания разработала набор математических моделей, описывающих зависимость распределения твердости от начального химического состава. Эти модели разработаны для определенных классов сочетаний диапазонов изменения концентрации химических компонентов.

Каждому классу соответствует свой набор моделей. Из всего возможного многообразия классов выделено 8 наиболее употребимых сочетаний, соответственно разработано 8 наборов моделей для решения проблем выбора начального химического состава.

При этом для прогноза твердости при выбранном для пробы химическом составе используется одна из моделей системы. Чаще всего оператору трудно подобрать модель, по которой лучше всего совершить прогноз. Поэтому возникает другая проблема выбора такой наиболее адекватной модели, которая обеспечивала наиболее точный прогноз.

Решением этой проблемы является использование универсальной модели Сугено (TSK), представляющей собой систему нечетких продукционных правил, обеспечивающей прогноз в виде линейной комбинации результатов всех известных 8 моделей.

Для этого было предложено разработать такое автоматизированное рабочее место, в котором все восемь моделей прогноза конечной твердости стали были бы объединены в одну.

Постановка задачи

Необходимо разработать автоматизированное рабочее место для прогнозирования твердости стали. Данная система должна выполнять следующие задачи:

- разработка модели для решения задач прогнозирования твердости стали;
- расчет прокаливаемости стали в модели TSK, с использованием всех известных 8 моделей;
- отображение графиков изменения твердости стали по ее глубине;
- разработка удобного интерфейса для оператора плавки.

Анализ предметной области

Для разработки модели прогнозирования твердости стали в виде нечетких продукционных правил (модель TSK) необходима фаззификация нечетких интервалов K_i^j (рис. 2) изменения массовой доли x_i i-х химических элементов. Правила для рассматриваемого случая записываются следующим образом:

$$\text{Если } \bar{x} \in K^j \quad c \quad \mu_{Kj}(\bar{x}) \text{ , то } \quad \bar{y}(h) = \bar{a}_0^j + A^j \bar{x} \quad c \quad \mu_{Kj}(\bar{x}) \qquad (1)$$

,где K_i^j - j-ый интервал изменения i - ого химического элемента;

$\mu_{Kji}(x_i)$ - значение функции принадлежности x_i соответствующему интервалу.

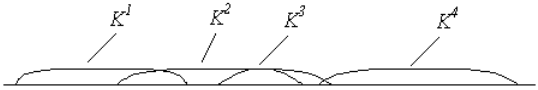

Рис. 2. Иллюстрация взаимного расположения нечетких интервалов изменения содержания химических элементов

Разбивка на интервалы (классы) осуществляются экспертами, исходя из соображений выделения наиболее часто встречающихся сочетаний значений содержания химических элементов для различных марок стали. Нечеткость таких интервалов порождается неполным соответствием между значениями содержаний химических элементов на различных участках интервалов и соответствующим линейным регрессионным уравнением. Как известно [1, 2], наилучшее соответствие на интервале достигается в точке математического ожидания выборочных значений химического состава. В условиях случайного формирования выборки предполагается, что наилучшее соответствие ($\mu_{K_i^j}(M(x_i)) = 1$) будет достигаться в середине интервала и монотонно убывать к его границам. В работе предлагается использовать для построения функций принадлежности $\mu_{K_i^j}(x_i)$ треугольные нечеткие числа [3, 4].

Для треугольных функций принадлежности имеется три определяющих параметра: левая и правая границы интервала и точка высоты треугольника (рис. 3).

Максимальное значение принадлежности (высота треугольника $\mu_{K_i^j}(x_i) = 1$) находится в центре табличного интервала [4].

После того, как определяется химический состав стали, производится расчет коэффициентов принадлежности с использованием нечетких продукционных правил (1)

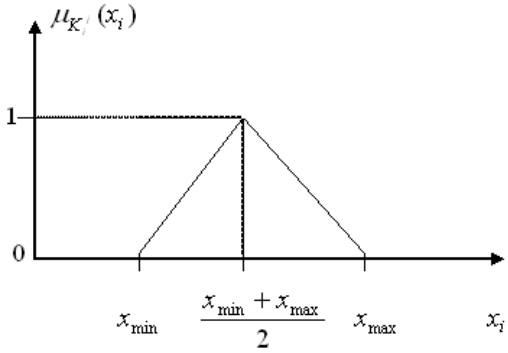

Рис. 3. Графическая иллюстрация параметров треугольного числа

В работе для полученных результатов функции принадлежности для каждого класса используется следующая операция

$$\mu_{Kj}(\bar{x}) = \min\{\mu_{K_i^j}(x_i)\} \tag{2}$$

Полученные выражения позволяют строить модель TSK для оценки конечной твердости стали. Прогнозное значение каждой компоненты вектора $\bar{y}(h) = (y_1,...,y_k)$ определяется по формуле

$$y_i = \frac{\sum_{j=1}^{8} \mu_{Kj}(\bar{x}) y_i^{\,j}}{\sum_{j=1}^{8} \mu_{Kj}(\bar{x})}, \text{ для } i = 1,\ldots,k. \tag{3}$$

Как видно, модель TSK снимает проблему выбора наиболее адекватной модели расчета твердости стали по химическому составу из известных восьми регрессионных моделей, т.к. позволяет включать в расчет прогнозного значения твердости стали все восемь моделей с весами, определяемыми значениями соответствующих функций принадлежности.

Реализация

Общий вид АРМ представлен на рис. 4

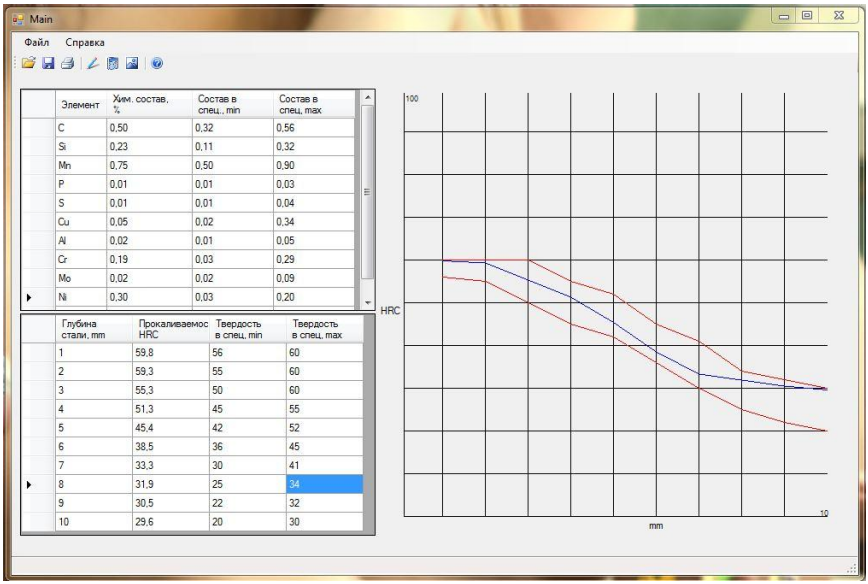

Рис. 4. Интерфейс автоматизированной рабочего места

Основная форма имеет две таблицы:

а) таблица химических элементов, выбранный состав и состав в спецификации;

б) таблица значений твердости стали по модели TSK, глубина стали, твердость стали заданная в спецификации.

Построение графиков осуществляется путем выбора в меню пункта “Построить”. Для того чтобы рассчитать прокаливаемость стали в модели TSK (по всем 8 моделям) выбирается пункт “Рассчитать прокаливаемость стали”, после чего оператор переходит на форму расчетов (Рис. 5).

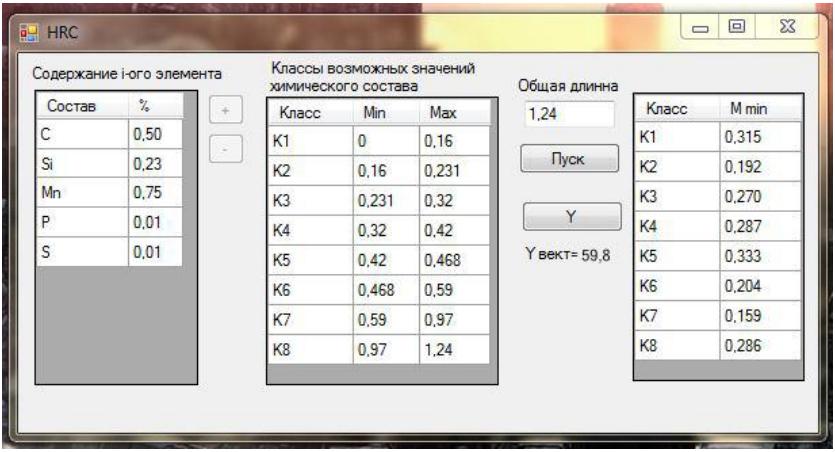

Рис. 5. Интерфейс формы расчетов по модели TSK

Форма расчетов имеет две основные таблицы:

- содержание i-го элемента;
- классы возможных значений химического состава.

Также форма расчетов содержит таблицу результатов минимальных функций принадлежности для каждого класса, кнопки запуска подсчетов μ_{min} и $\overline{y}$.

Заключение

В данной статье была описана разработка автоматизированного рабочего места, обеспечивающая возможность расчета прокаливаемости стали в модели TSK, с использованием всех известных 8 моделей. Также было разработано отображение графиков изменения твердости стали по ее глубине и удобный интерфейс.

Список литературы

1. Бондарчук А.А. Модели управления твердостью металла в условиях стохастической и нечеткой неопределенности. / Бондарчук А.А., Матвеев М.Г., Полянский Ю.А. // Системы управления и информационные технологии. М.: № 4.1, 2007, стр.124-128.

2. Бобылев М.В. Оптимизация прокаливаемости и состава термоулучшаемой борсодержащей стали./ Бобылев М.В., Ламухин А.М., Кувшинников О.А. // Сталь, № 7, 2002. С. 68-71

3. Бондарчук А.А., Решение задач выбора при квазилинейных моделях объекта управления. / Бондарчук А.А., Матвеев М.Г. // Современные проблемы информатизации в анализе и синтезе программных и телекоммуникационных систем. Сб. трудов. Вып.13. Воронеж: Научная книга, 2008, стр. 283-285.

4. Филипчук А.С. Интенсификация плавления металлизированных окатышей ДСП. / Филипчук А.С., Бондарчук А.А., Меркер Э.Э., Кожухов А.А. // Материалы международной научно-технической конференции «Азовсталь - 2005». Мариуполь, 2005, стр. 26-27.

ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ПРОГНОЗИРОВАНИЯ СТОИМОСТИ НЕДВИЖИМОСТИ

студ. Эльда Джумари,
проф. А. А. Сирота

Введение

Нейронные сети – это одно из направлений исследований в области искусственного интеллекта, основанное на попытках воспроизвести нервную систему человека. А именно: способность нервной системы обучаться и исправлять ошибки, что должно позволить смоделировать, хотя и достаточно грубо, работу человеческого мозга [1].

Важнейшая особенность сети, свидетельствующая о ее широких возможностях и огромном потенциале, состоит в параллельной обработке информации всеми звеньями, что позволяет значительно ускорить процесс обработки информации. Кроме того, при большом числе межнейронных

соединений сеть приобретает устойчивость к ошибкам, возникающим на некоторых линиях.

В настоящее время нейронные сети используются для решения целого ряда задач, одной из которых является задача прогнозирования.

Прогнозирование – это предсказание будущих событий. Пусть заданы n дискретных отсчетов $\{y(t_1), y(t_2), ..., y(t_n)\}$ в последовательные моменты времени $t_1, t_2, ..., t_n$. Тогда задача прогнозирования состоит в предсказании значения $y(t_{n+1})$ в некоторый будущий момент времени t_{n+1}.

Целью прогнозирования является уменьшение риска при принятии решений. Прогноз обычно получается ошибочным, но ошибка зависит от используемой прогнозирующей системы. Предоставляя прогнозу больше ресурсов, можно увеличить точность прогноза и уменьшить убытки, связанные с неопределенностью при принятии решений. Типичными приложениями техники прогноза являются предсказание цен на фондовой бирже, прогноз погоды, прогноз потребления электроэнергии, прогноз отказов технических систем и т.д.

В данной статье рассматривается применение нейронной сети для решения задачи прогнозирования стоимости недвижимости.

Объектом исследования является рынок продажи жилой недвижимости города Воронежа.

Целью данной работы является разработка методов прогнозирования модели оценки изменения цен реальных сделок на рынке продажи жилого фонда недвижимости, основанной на нейросетевых технологиях и позволяющей существенно повысить эффективность работы организации, занимающейся продажей жилья.

На начальном этапе проводился анализ известных методов прогнозирования данных с использованием искусственных нейронных сетей.

Вторая задача заключалась в разработке собственной модели и нейросетевого алгоритма стоимости недвижимости, показывающего высокую производительность при решении задач большой размерности.

Третья задача – разработка программы и исследование алгоритма стоимости недвижимости.

При решении указанных задач использовались современные методы теории искусственных нейронных сетей, градиентные методы оптимизации, возможности нейросетевого пакета среды Matlab [2].

Методика эксперимента

Для решении перечисляемых задач проводилось несколько экспериментов. Целью экспериментов было прогнозирование стоимости недвижимости в городе Воронеж. Данные использовалось с 1 октября 2012 года по 15 марта 2013 года.

Каждый из экспериментов можно разбить на несколько этапов. Первым этапом было формирование обучающей выборки. На этом этапе определяется вид представления данных и происходит формирование наборов, подаваемых на входные нейроны и соответствующие им наборы снимаемых с выходов сети.

На входе сети подаются данные, которые зависят от 5 факторов: район, этаж, тип, количество комнат, жилая площадь. На выходе сети получаем цену.

Исходные данные преобразуются к виду, в котором их можно подать на входы сети. В рассматриваемом примере есть количественные (этаж, количество комнат, жилая площадь) и неколичественные данные (район, тип). Неколичественные данные сначало пронумеруем, а дальше все данные нормируем таким образом: пусть $x = \{x_1, x_2, ... x_n\}$ - множество каждого фактора, которого нужно пронормировать, при $n = \overline{1,i}$, i - количество элементов множества x (в нашем примере i =100), тогда:

$$x_{norm} = \frac{x_i}{x_{\max}},$$

где $x_{\max}$ - максимальное значение множество x, x_{norm} - нормируемое значение. В результате получаем данные в диапазоне [0,1].

В качестве архитектуры сети было решено выбрать многослойную нейронную сеть прямого распространения [3], так как ее можно успешно применить для решения большинства задач.

Первый эксперимент - прогнозирование данных с использованием аппарата искусственных нейронных сетей. Для построения прогнозирующего алгоритма проводилось обучение и тестирование (где тестирование производится на обучающих данных) нейронной сети прямого распространения (нейронной сети типа «многослойноый персптрон»).

Тестирование качества обучения нейросети необходимо проводить на примерах, которые не участвовали в ее обучении. При этом число тестовых примеров должно быть тем больше, чем выше качество обучения.

Во втором эксперименте осуществлялось прогнозирование данных с использованием аппарата искусственных нейронных сетей. Для построения прогнозирующего алгоритма проводилось обучение и тестирование (где тестирование производится на данных, не используемых при обучении).

Результаты в этом случае чуть хуже, чем во время первого эксперимента.

В таблице 1 показаны результаты экспериментов при разных параметрах архитектуры.

Таблица 1 - Результаты экспериментов

Параметры архитектуры	СКО для эксперимента 1				СКО для эксперимента 2			
	Количество эпох				Количество эпох			
Нейроны в скрытом слое	5	100	1000	10000	5	100	1000	10000
5	0,2062	0,1507	0,1567	0,1577	0,2226	0,1512	0,1644	0,1716
10	0,2044	0,1551	0,1593	0,1615	0,3007	0,1638	0,1791	0,1836
20	0,3618	0,1590	0,1609	0,1627	0,3959	0,1793	0,1913	0,1968

В левой колонке указаны параметры архитектуры - нейроны в скрытом слое. В верхней строке – среднеквадратичная ошибка для каждого эксперимента при разных количество шагов, отображено в колонках.

Результаты экспериментов показывают, что для эффективного обучения сети достаточно 100…1000 эпох. При этом количество нейронов в скрытом слое достаточно выбрать порядка 5…10. Некоторое увеличение ошибки при увеличении числа нейронов и количества эпох объясняется недостаточным объемом обучающих данных, что приводит к возникновению эффекта переобучения. В целом, общая средняя квадратичная ошибка прогнозирования стоимости недвижимости составила 15 - 20%.

Выводы

По результатам выполненной работы можно сделать следующие выводы:

1) Нейронные сети являются очень мощным инструментом для работы на финансовых рынках, но для освоения технологии нужно потратить времени уже точно не меньше, чем на освоение технического анализа.

2) Основные проблемы, возникающие при работе с этой технологией – правильная предобработка данных, этот этап играет решающую роль для прогнозирования данных.

3) Среднеквадратическая относительная ошибка прогноза по модели составляет примерно 20%. Построенная модель позволяет повысить эффективность управления комплексами недвижимости в масштабах города или крупной корпорации и сделать этот механизм более прозрачным.

Список литературы

1. Осовский С. Нейронные сети для обработки информации / С. Осовский. – М.: Финансы и статистика, 2002. – 343 с.

2. Дьяконов В. Математические пакеты расширения MATLAB. Специальный справочник / В. Дьяконов, В. Круглов. – СПб.: Питер, 2001. – 480 с.

3. Кирсанов Э. А. Обработка информации в пространственно-распределенных системах радиомониторинга: статистический и нейросетевой подходы / Э. А. Кирсанов, А. А. Сирота. – М.: Физматлит, 2012. – 343 с.

4. Зиновьев А. Ю. Визуализация многомерных данных / А. Ю. Зиновьев. – Красноярск, 2000. – 180 с.

РЕАЛИЗАЦИЯ БЕСПРОВОДНОЙ СЕТИ НА ОСНОВЕ МИКРОКОНТРОЛЛЕРА MSP430

студ. А.С. Дущенко,
доц. Д.Н. Борисов

В последнее время беспроводные устройства связи все больше вытесняют проводную связь. С одной стороны это связано с повышением интеграции и уменьшением стоимости радиочастотных компонентов — радиотрансиверов, антенн и т. п., а с другой — с ростом стоимости материалов и трудозатрат для организации проводной связи. Если ранее эта тенденция была заметна при дальности связи, исчисляемой километрами, то сейчас бум беспроводных технологий приходит в область устройств локальной связи с дальностью от единиц до нескольких сотен метров. Даже при таких небольших расстояниях суммарная стоимость материалов и комплектующих, а также работ по монтажу линии связи может оказаться существенно выше, чем применение беспроводных сетей. Кроме того, немаловажным является вопрос энергопотребления устройств работающих в беспроводных сетях. Разрабатываются и реализуются различные стандарты и устройства, позволяющие значительно снизить потребление энергии при работе с беспроводными сетями. Таким образом, вопросы, связанные с реализацией беспроводных сетей и имеющих низкое энергопотребление различных, в том числе мобильных устройств, является крайне актуальным.

1. Постановка задачи

Реализация беспроводной сети с использованием микроконтроллера MSP430 и радиотрансивера CC2500 состоит из следующих задач:

- иметь возможность измерения температуры окружающей среды;
- давать возможность просмотра вольтажа используемых устройств;
- предоставлять возможность просмотра данных на персональном компьютере.

2. Анализ предметной области

Для возможности выполнения данной задачи были выбраны следующие решения.

Протокол SimpliciTI, разработанный Texas Instruments в качестве простой альтернативы ZigBee для реализации небольших беспроводных сетей с числом абонентов менее 256. В первую очередь протокол ориентирован на сети с ограниченным числом узлов с батарейным питанием.

Стек SimpliciTI включает в себя несколько уровней, иерархия которых показана на рис. 1.

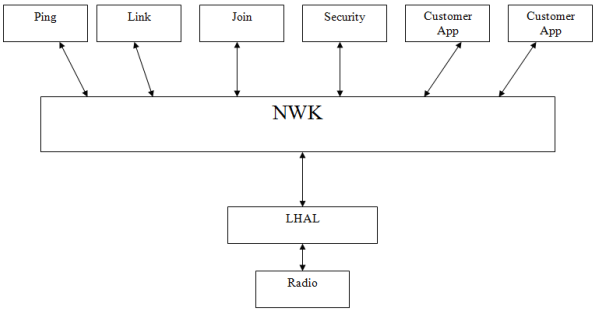

Рис. 1. Стек протокола

Самый нижний уровень — Radio — реализует физический уровень связи. По сути это сам интегральный трансивер, CC2500. Уровень LHAL позволяет абстрагироваться от аппаратных особенностей интерфейса SPI к более высокому уровню — NWK, обращающемуся к Radio. Уровень NWK определяет передачу кадров и управление работой сети, включая процессы соединения, установки частот и каналов, работу с ключами шифрования, а также управление функционированием других элементов сети. На данном уровне работают основные алгоритмы стека, в которых реализуется управление приемом и передачей пакетов и кадров. Соединение организуется путем открытия портов, аналогичных портам стека TCP/IP. Соединения различных типов обращаются к NWK через определенные порты [1].

Вторым решением для реализации беспроводной сети является инструмент eZ430-RF2500 (рис. 2) для проектирования встроенных систем, в которых объединяются микроконтроллеры MSP430 со сверхнизким энергопотреблением (MCU) и системы беспроводной связи.

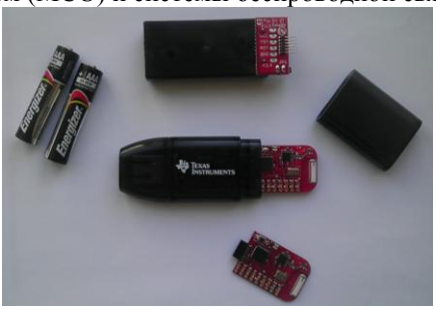

Рис. 2. Инструмент eZ430-RF2500

Инструмент с интерфейсом USB предлагает разработчику две отладочные платы под микроконтроллеры с поддержкой радиочастотной (RF) связи, а также компьютерный отладочный интерфейс, который можно использовать для разработки проектов автономных беспроводных приложений.

В отладочную плату eZ430-RF2500T интегрированы микроконтроллер MSP430F2274 и радиотрансивер CC2500, разработанные компанией TI. Микроконтроллер MSP430F2274, основанный на 16-разрядной RISC-архитектуре, может обрабатывать значительные объемы программного кода и данных и имеет 32 Килобайта флэш-памяти и 1 килобайт оперативной памяти [2].

3. Реализация

Общая топология беспроводной сети представлена на рис. 3.

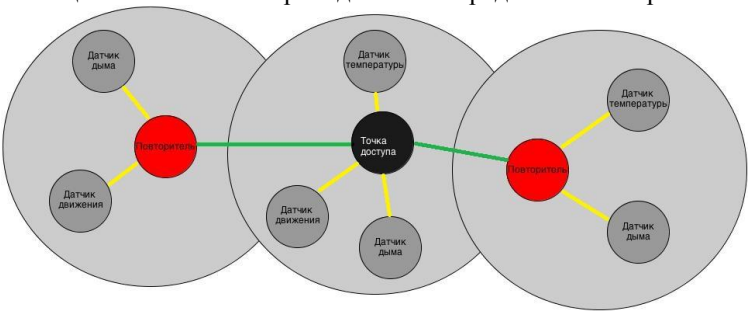

Рис. 3. Топология беспроводной сети

Устройства в сети могут быть 3х типов: Access Point, Range Extender и End Device, соответственно они выполняют различные функции. Устройство типа Access Point («Точка доступа», далее AP), выполняет функции управления сетью и адресами хранения сообщений для «спящих» устройств в сети. В одной сети может существовать только одна точка доступа. Устройства типа AP не могут быть переведены в спящий режим, и их питание рекомендуется выполнять стационарным с резервированием от батарей. Отметим, что устройства типа AP могут быть одновременно сконфигурированы и как устройства типа Range Extender или End Device.

Устройства типа Range Extender («Расширитель», далее RE), выполняют функции маршрутизации и повторителя и, так же как AP, требуют стационарного питания с резервированием, поскольку не предназначены для перевода в спящий режим.

Устройства типа End Device («Конечное устройство», далее ED), служат собственно для приема и передачи данных. Так как это устройства

батарейным питанием, то подразумевается, что устройства типа ED большую часть времени находятся в спящем режиме.

Выполнение программы (рис. 4) начинается с системной инициализации, которая практически идентична для точки доступа и конечных устройств сети.

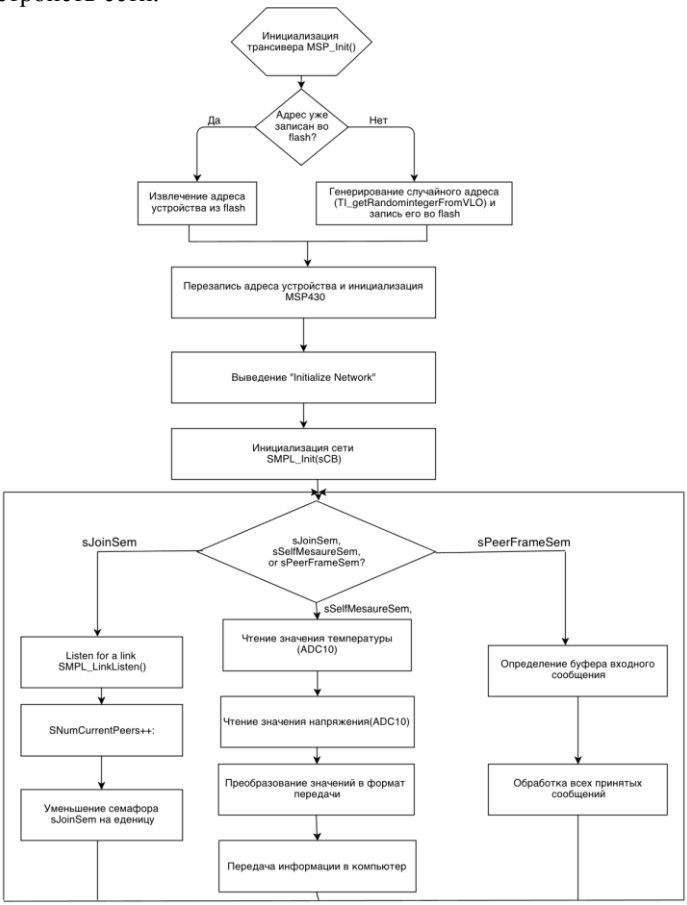

Рис. 4. Алгоритм выполнения программы

Функция MSP _ Init() определяет взаимодействие между MSP430 и CC2500. После аппаратной инициализации AP и ED создают (случайным образом) 4-байтный адрес, записывают этот адрес во flash-память. После завершения инициализации аппаратной части, вызывается функция инициализации сети SMPL _ Init(s C B). Параметр sCB представляет собой

указатель на функцию вызова, выполняющуюся в обработчике прерываний (ISR) при получении посылки точкой доступа. В соответствии с идентификатором канала связи, функция обратного вызова sCB определяет и инкрементирует соответствующий семафор sPeerFrameSem или sJoinSem для их обработки в теле основной программы. В коде программы точки доступа также используется семафор sSelfMeasureSem, который устанавливается таймером A, чтобы каждую секунду измерять и отображать собственные значения температуры и напряжения точки доступа. Данный алгоритм полностью описывает работу точки доступа.

Заключение

В данной работе описано развертывание беспроводной сети на основе микроконтроллера MSP430 и радиотрансивера CC2500, обеспечивающая возможность просмотра температур с датчиков радиотрансивера, работающая на основе протокола беспроводной передачи данных SimplisiTI.

Список литературы

1. Протокол беспроводной передачи данных SimplisiTI (URL – http://www.ti.com/tool/simpliciti).
2. Семенов Б.Ю. Микроконтроллеры MSP430:Первое знакомство / Б.Ю. Семенов. – М.: Изд-во: «Солон-пресс», 2006. – 126 с.

МОДУЛЬ АВТОМАТИЧЕСКОГО ПЛАНИРОВАНИЯ РАБОЧЕГО ВРЕМЕНИ МЕНЕДЖЕРОВ БАНКА НА ПЛАТФОРМЕ ORACLE SIEBEL CRM

студ. И. М. Дюжева,
ст. преп. Д. И. Соломатин

В условиях постоянно возрастающей конкуренции на рынке товаров и услуг все более актуальной становится задача персональной работы с потребителем. В связи с активным развитием банковского сектора в России, существует проблема привлечения новых и удержания старых клиентов из-за ужесточения конкурентной борьбы между банками. Для решения этой проблемы создаются специальные отделы, призванные учитывать интересы потребителей.

В банках индивидуальной работой с клиентами занимаются сотрудники на должности менеджеров по развитию бизнеса (МРБ). Они, в соответствии со стратегией компании или по назначенному руководителем плану, выезжают к клиенту, проводят переговоры с партнерами (торговые

организации, первые лица) и показ презентаций, повышают уровень взаимного доверия, осуществляют обучение персонала, контролируют продажи партнеров, ведут отчетность для руководства.

При этом каждый руководитель должен планировать работу своих менеджеров. В сферу его деятельности входит определение плановых финансовых показателей, которых необходимо достичь менеджеру при работе с клиентами для выполнения плана продаж. Также одной из задач менеджера является оптимизация своего рабочего времени, которая связана с уменьшением издержек на написание недельного плана работы и поиска маршрута проездов к клиенту. В компании должна быть единая электронная форма по планированию рабочего дня менеджера с автоматическим формированием отчета.

Таким образом, стала актуальной задача автоматизации рутинной работы менеджеров. Целью данной работы являлась разработка модуля планирования времени со следующими возможностями:

- Выполнение расчета стратегий посещений менеджерами торговых точек (ТТ);
- Формирование записей о визитах на каждый рабочий день менеджера в соответствии с примененными стратегиями;
- Выбор оптимального маршрута в соответствии с планом текущего рабочего дня;
- Составление ежедневных/еженедельных/ежемесячных отчетов о рабочем времени для руководителей.

При реализации вышеуказанного модуля были решены следующие задачи:

- Составление алгоритма формирования визитов по заданным стратегиям;
- Интеграция Oracle Siebel CRM с геоинформационной системой банка и системой анализа и отчетности «Oracle BI Publisher».

Под CRM понимается модель взаимодействия, полагающая, что центром всей структуры бизнеса является клиент, а основными направлениями деятельности являются меры по поддержке эффективного маркетинга, продаж и обслуживания клиентов [1].

К основным принципам CRM относятся:

- наличие единого хранилища информации;
- использование всех каналов взаимодействия – телефонные звонки, регистрационные формы на web-сайтах, рекламные ссылки, системы корпоративного web-чата и т. д.;
- постоянный анализ собранной информации о клиентах.

Структура данных разрабатываемой системы представлена на рис. 1.

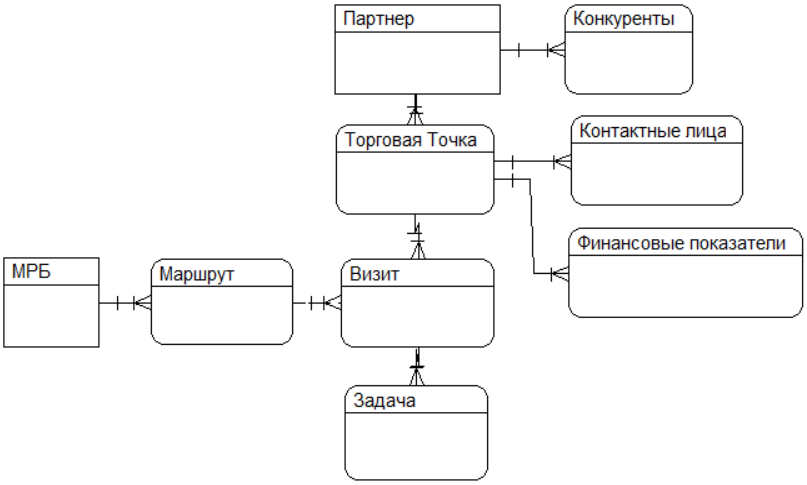

Рис. 1. Схема БД приложения

Стратегия посещений менеджерами ТТ предполагает набор следующих параметров:

1) категория ТТ (A, B, C);
2) статус ТТ (активная, потенциальная, продающая);
3) доли визитов по статусам стратегии ТТ;
4) доли ТТ по категориям;
5) доля визитов по «пожарам»;
6) доля визитов по свободному времени;
7) лимит визитов – количество визитов на одного МРБ за 4 недели.

Алгоритм расчета количества ТТ при заданной стратегии состоит из следующих шагов:

1) задается максимальный потенциал ТТ;
2) в соответствии с процентным соотношением визитов по статусам ТТ высчитывается количество продающих ТТ;
3) среди всех Продающих ТТ в соответствии с процентным соотношением визитов по категории рассчитывается количество визитов по категориям A, B, C;
4) переход к шагу 2 и проход по всем Активным ТТ;
5) переход к шагу 4 и проход по всем Потенциальным ТТ.

Порядок формирования автоматического планирования визитов в системе:

- Система ежедневно по расписанию проверяет на всех карточках ТТ наличие нового загруженного признака «Пожар». Если признак «Пожар» появился, то система в тот же день по расписанию осуществляет формирование Визита по событию «Пожар»;
- каждую пятницу по расписанию Система (в 20:00) автоматически запускает процедуру планирования Визитов по кратности;
- каждую пятницу после завершения отработки процедуры планирования Визитов по кратности, Система автоматически запускает процедуру планирования визитов по Пожарам.

Построение отчетов вызывается в системе встроенным интеграционным workflow-процессом. Данные из БД выгружаются сначала в xml-файлы, а затем xml-файлы посредством xslt-преобразований трансформируются в excel-документы.

После формирования записей о визитах создается список посещаемых ТТ с адресами по каждой из них. Важно выбрать оптимальный маршрут, чтобы тратилось минимальное время на дорогу. Таким образом, в разрабатываемом модуле была использована задача коммивояжёра с целью выбора минимального по времени поездки маршрута. Задача коммивояжёра – одна из самых известных задач комбинаторной оптимизации, заключающаяся в отыскании самого выгодного маршрута, проходящего через указанные города хотя бы по одному разу с последующим возвратом в исходный город. Критерием выгодности маршрута является минимальное время поездки [2]**.** Время поездки рассчитывается исходя из расстояния до ТТ, а также средней скоростью в это время года и суток.

Внедрение модуля в промышленную эксплуатацию позволило снизить нагрузку на руководителей менеджеров посредством удобного механизма планирования и контроля выполнения посещений ТТ, увеличить эффективность работы менеджеров за счет планирования ежедневной деятельности и разработки оптимального маршрута посещения точек продаж, а также быстро реагировать на изменение ситуации в разных торговых точках.

Список литературы

1. Джилл Д. CRM-Навигатор. Пособие по управлению взаимоотношениями с клиентами / Д. Джилл. – М.: Библиотека Profix, 2008. - 725 с.

2. Хаггарти Р. Дискретная математика для программистов / Р. Хаггарти; пер. с англ., под ред. С.А. Кулешова, с доп. А.А. Ковалева. – М.: Техносфера, 2003. – 320 с.

РАЗРАБОТКА СИСТЕМЫ ДЛЯ РАСЧЕТА МЕТРИК КАЧЕСТВА КОДА И ИССЛЕДОВАНИЕ ИХ ЗНАЧИМОСТИ

студ. И. В. Иванников,
ст. преп. Д. И. Соломатин

1. Метрики качества программы

Чем отличается “плохая” программа от “хорошей”? С точки зрения конечного пользователя качество системы определяется функциональными возможностями и надежностью. Однако для разработчика на первое место по значимости выходят характеристики исходного кода программы, отражающие количественные показатели и логическую сложность системы. Эти данные позволяют выявить участки кода, для которых необходимо провести рефакторинг, а также контролировать общий уровень качества программного обеспечения (ПО) на всех этапах разработки.

Для контроля и оценки качества служат метрики кода. Метрика программного обеспечения (software metric) – численное значение, отражающее состояние определенного свойства системы. Они позволяют оценить характеристики кода по различным критериям: структурная сложность, связанность компонентов, их объем и др. Значения метрик вычисляются путем анализа исходного текста программы.

Однако главная сложность состоит в том, что показания метрик – это всего лишь числа, интерпретация и оценка которых ложится на плечи разработчика. Для этого существуют различные метрические шкалы и критерии оценки. Но решающее значение имеет контекст анализируемой системы: сложность решаемой задачи, используемые программные средства, профессиональный уровень разработчика, значения параметров аналогичных программ и т.д. Как правило, качество ПО оценивается по совокупности нескольких метрик, т.к. отдельно взятые параметры могут не отражать интересующие характеристики или иметь случайный характер.

Для оценки значимости было проведено исследование 3-х метрик: Холстеда, Т. Джилба и Чепина. Они относятся к 3-м основным классам метрик: количественные, сложности потока управления программы и сложности потока управления данными, соответственно.

Для проведения эксперимента была разработана библиотека, реализующая расчет этих параметров для программ, написанных на языке C# [1]. В ней

использовался статический анализ исходных текстов приложений [2], метод конечных автоматов и регулярные выражения. Это позволило избежать жесткой привязки к языку разработки ПО. Для анализа программы написанной на другом языке достаточно заменить ключевые конструкции отражающие синтаксис.

2. Метрика Холстеда

Основу метрики Холстеда [3] составляют четыре измеряемые характеристики программы: n_1 - число уникальных операторов программы, включая символы-разделители, имена функций и знаки операций (словарь операторов); n_2 - число уникальных операндов программы (словарь операндов); N_1 - общее число операторов в программе N_2 - общее число операндов в программе.

На их основе вводятся следующие оценки:

- словарь программы (n=n1+n2) – это тот объем средств языка разработки, который использовался при написании программы, из всего множества средств (количество уникальных операторов, количество использованных типов переменных и т.д.);
- длина программы (N=N1+N2) – отражает количественную величину использованных средств языка, т.е. то, как часто использовались те или иные конструкции языка, сколько введено переменных каждого типа и т.д.;
- объем программы (V=Nlog2n) – определяет соотношение использованных средств языка, к объему, в котором они использовались.

Объем программы является более объективной характеристикой размера программы, чем количество строк кода - SLOC (source lines of code).

3. Метрика Т. Джилба

Метрика Т. Джилба [4] является одной из наиболее простых, но достаточно эффективных оценок сложности управления программ. Ее суть состоит в определении логической сложности программы, как насыщенность кода конструкциями IF_THEN_ELSE. При этом вводятся две характеристики:

- CL - абсолютная сложность программы, отражает количество операторов условия;
- cl - относительная сложность программы, характеризует насыщенность программы операторами условия и определяется как отношение CL к общему числу операторов.

Несмотря на свою простоту, результирующее значение этой метрики позволяет оценить сложность программы, с точки зрения алгоритмической сложности.

4. Метрика Чепина

Значение метрики Чепина [4] отражает информационную прочность отдельно взятого программного модуля с помощью анализа используемых в его коде переменных, выполняющих различные функции: управление циклами и ветвлением, передача и прием параметров, организация вывода и т.д.

Исходя из этой концепции, на основе информации о количестве переменных, выполняющих определенную роль в программе, можно судить о насыщенности кода определенными конструкциями и приемами программирования. Это, в свою очередь, позволяет оценить сложность текста программы по различным критериям.

Все множество переменных, объявленных в тексте программы, разбивается на 4-е функциональные группы:

- P - переменные, организующие ввод информации и используемые в расчетах. Примером могут служить параметры функций;
- M - модифицируемые, или создаваемые внутри программы переменные;
- C - переменные, участвующие в управлении работой программного модуля - управляющие переменные циклов, условных операторов и т.д.;
- T - не используемые в программе, так называемые “паразитные”, переменные.

Описанные величины отражают количество переменных, входящих в каждую группу. Поскольку каждая переменная может выполнять одновременно несколько функций, необходимо учитывать ее в каждой соответствующей функциональной группе.

Результирующее значение этой метрики обозначается Q и рассчитывается по формуле:

$Q = a1*P + a2*M + a3*C + a4*T,$

где a1, a2, a3, a4 — весовые коэффициенты.

Весовые коэффициенты отражают различное влияние на сложность программы каждой функциональной группы. Наибольший вес, равный 3-м, имеет функциональная группа С, так как она влияет на поток управления программы. Весовые коэффициенты остальных групп распределяются следующим образом: a1=1, a2=2, a4=0.5. Весовой коэффициент группы Т не равен 0, поскольку “паразитные” переменные не увеличивают сложность потока данных программы, но иногда затрудняют ее понимание. С учетом весовых коэффициентов:

$$Q = P + 2M + 3C + 0.5T$$

Это эффективный способ оценить сложность программы с точки зрения использования данных, которыми оперирует приложение, т.к. подобная информация заложена, как правило, именно в переменных.

Подобные метрики являются наиболее распространенными и востребованными при оценке программного обеспечения. Они выражают значения характеристик, имеющих большое значение при анализе и прогнозировании сложности поддержки ПО. На основе этих показателей принимаются решения о рефакторинге кода, а конкретно тех его участков, для которых они принимают негативные краевые значения. Допустимые границы колебаний параметров определяются на основании контекста решаемой задачи, опираясь на значения других, как правило, количественных метрик.

5. Проведение эксперимента и обоснование полученных результатов

Для проведения эксперимента была сформирована тестовая выборка из 50-и программ, включающих приложения написанные студентами младших (1-3) и старших (4-6, магистратура) курсов в равной пропорции. Они были названы 1-а и 2-а группы, соответственно. Для них были рассчитаны описанные выше метрики, а также значения параметров, которые позволяют вычислить встроенные инструменты среды разработки: общий индекс сложности поддержки кода, цикломатическая сложность, глубина наследования, связанность классов и количество строк кода, которые являются наиболее значимыми характеристиками при анализе качества программы. Это необходимо для рассмотрения данных о системе в совокупности.

Полученные данные были систематизированы. Каждая метрика была исследована на наличие связей с другими параметрами кода. Определена значимость каждой отдельно взятой характеристики. Анализ полученных данных показал следующие результаты.

В программах объем до 150, по метрике Холстеда, разработчики используют большое количество вспомогательных переменных, не заботясь об оптимизации кода и вычислений. Это связанно с небольшой длинной текста и сложностью решаемых задач. Однако в диапазон 150-170 попадает большая группа проектов, для которых при увеличении объема, длинна кода практически не изменяется. Скорее всего, это обусловлено тем, что после увеличения длинны программы до определенного размера, разработчик “искусственно” уменьшает его, используя сокращенные формы записи и более компактные конструкции языка. Это способствует визуальному уменьшению длинны кода, и более удобному восприятию текста программы. Подобные шаблоны в распределении значений указанных параметров встречаются и в более объемных программах. Это свидетельствует о том, что существует определенная длинна классов и функций, относительно объема решаемых задач, которая удобна для визуального восприятия. После превышения локального порогового значения появляется необходимость рефакторинга кода, для улучшения восприятия.

Сложность потока управления данными в программах студентов начальных курсов обратно пропорциональна структурной сложности кода, что вкупе с увеличением длинны программы свидетельствует о более линейном ходе выполнения приложения. Это может быть признаком неоптимального использования циклических конструкций. В программах студентов старших курсов сложность управления данными не возрастает с увеличением циклической сложности, притом, что увеличивается объем и длинна кода. Это является следствием использования сложных алгоритмических структур для обработки данных. Так же, это косвенно может указывать на уместное использование локальных переменных в циклах.

Значение метрики Джилба для всех программ из тестовой выборки колеблется в диапазоне 0-0,2. Связь с другими измеренными характеристиками для нее не выявлена. Но, исходя из методики расчета этого параметра, можно предположить, что значение этой метрики остается неизменным, так как пропорционально увеличению сложности решаемых задач возрастает потребность в использовании большего числа операторов, в том числе и условных.

Список литературы

1. Официальная документация по C# и Microsoft Visual Studio http://msdn.microsoft.com

2. Рыжков Е. Программный код и его метрики. Блог компании Intel. http://habrahabr.ru/company/intel/blog/106082/
3. Новичков А. Метрики кода. http://anovichkov.msk.ru/?p=1196
4. Статический анализ кода. http://www.viva64.com/ru/t/0046/.

АЛГОРИТМ РАСПОЗНАВАНИЯ РУКОПИСНЫХ СИМВОЛОВ НА ОСНОВЕ АНАЛИЗА НАПРАВЛЕНИЯ ПЕРЕМЕЩЕНИЯ ПЕРА

студ. Э.Э. Киселёв,
проф. С.А. Запрягаев

Задача распознавания рукописного текста и символов является предметом исследования в течение длительного периода времени. Сложность ее решения определяется широким диапазоном изменения свойств объектов, подлежащих распознаванию. Набор методов решения такой задачи достаточно широк и не универсален [1]. В связи с этим, разработка инструментов анализа различных методов распознавания для сравнения их характеристик является актуальной задачей и способствует поиску адекватных решений.

Цель настоящей работы состоит в разработке алгоритмов для анализа информации об изменении координат точек изображения объекта с течением времени.

В настоящем проекте разработана программная оболочка, которая фиксирует координаты пера (x, y) и преобразует их в параметрические плоские кривые. Исходно эти кривые имеют вид последовательности точек, которые дискретно отбираются цифровым пером в режиме реального времени

$$s = f\{p_i\} = f(x_i, y_i), i = 1,...,n \qquad (1)$$

где $f(x_i, y_i)$ – декартовы координаты точек пера на плоскости. На рис.1, 2, для примера, представлены кривые соответствующие условным, ненормированным координатам x_i и y_i (горизонтальная и вертикальная оси изображения) цифр от 0 до 9, в системе координат, в которой в качестве начальной точки выбрана исходная точка написания цифр. Кроме значений координат точек в различных устройствах, фиксируются условные величины давления пера на плоскость, а также наклон пера, что существенно расширяет возможности распознавания рукописного символа. По совокупности этих параметров можно построить различные

алгоритмы распознавания, которые могут эффективно применяться в процессе ввода информации [2].

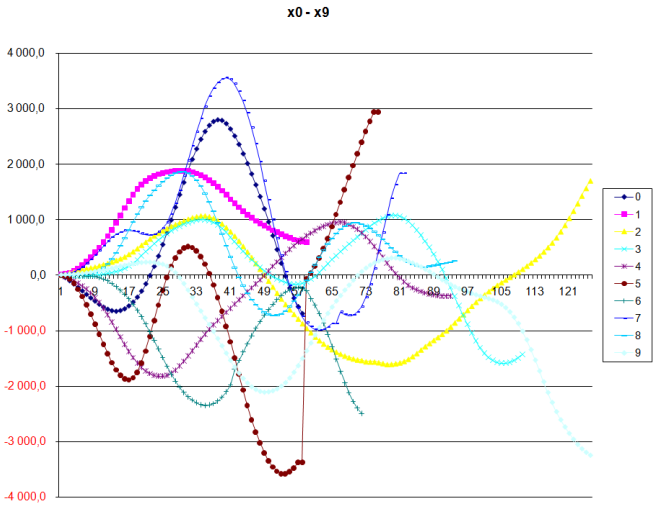

Рис. 1. Условные координаты x_i при написании цифр от 0 до 9

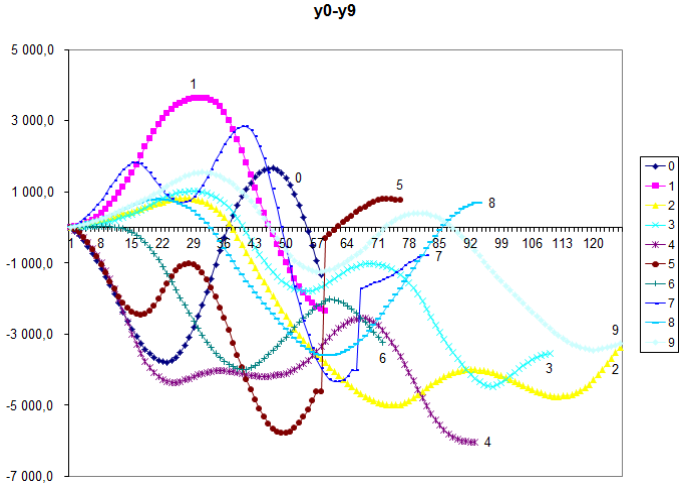

Рис. 2. Условные координаты y_i при написании цифр от 0 до 9

Как видно из рис. 1, 2, кривые, определяющие зависимость фиксируемых параметров от времени, для различных символов содержат достаточно обширный объем информации, который может быть использован для построения и классификаторов, и векторов признаков [3].

Программная оболочка

Разработанная в проекте программная оболочка обеспечивает отображение зависимости параметров x и y от времени во время ввода символа, а также обрабатывает эти последовательности данных путем анализа свойств кривых.

Для реализации построения классификатора, основанного на сравнении введенного символа с символами-эталонами, существует возможность подключения различного набора образцов рукописных символов из текстовых файлов, содержащих значения x и y.

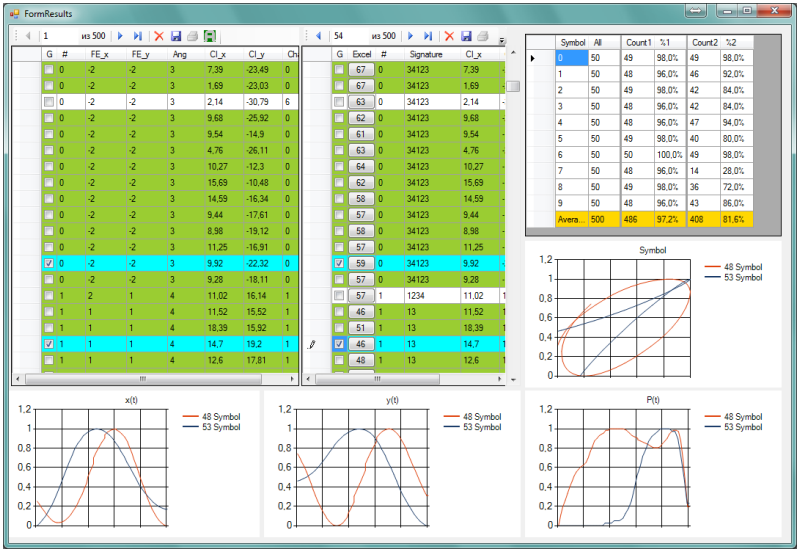

Рис. 3. Результат пакетной обработки символов.

В качестве основной функции разработанного программного обеспечения реализована пакетная обработка введенного массива символов. Общее представление окна интерфейса представлено на рис.3. Программа обрабатывает каждый образ рукописного символа из всего массива символов, содержащихся в папке database в виде текстовых файлов образца, представленного на рис.4. Анализ символа осуществляется заданным набором правил и приводит результаты

обработки в виде статистической сборки, представленной данными об общем количестве обработанных символов, о количестве распознанных символов и об их процентном соотношении. Это способствует сравнению качества различных алгоритмов распознавания при наличии достаточно обширного набора признаков и образцов.

```
1.dta
This file includes 60 packets.

   TIME      BUTTONS     ORIENTATION       X       Y     PRESSURE
69889639         1           0            0      7733    4408    4
69889662         1           0            0      7683    4305    8
...
69890082         1           0            0      7349    3100    53
69890090         1           0            0      7252    2812    27
69890097         0           0            0      7252    2812    27
End of data
```

Рис. 4. Образец текстового файла

На рис.4 представлены результаты применения алгоритма распознавания к встроенной базе символов. Основным компонентом формы результатов является таблица, содержащая в себе информацию о каждом из символов. Полями таблицы являются выделенные из образца признаки. Выбрав режим «Анализ» в меню программы, можно получить аналитическое представление результатов распознавания каждого из символов реализованным методом (правильно детектированные символы в таблице выделяются цветом). Имеется возможность в режиме реального времени построить графики зависимости различных характеристик траектории пера от времени (координаты, давление) и изображение самого символа [4]. Это необходимо для визуального анализа разработчиком причин неудачного распознавания.

Алгоритм распознавания на основе параметров направления движения пера

Разработанная программная оболочка является вспомогательным инструментом для разработки и анализа различных алгоритмов распознавания рукописного текста. На рис.5 представлены образцы кириллических букв «а» и «н», введенных в программную оболочку.

Рис. 5. Образцы кириллических букв «а» и «н»

Ненормированные координаты и давление, выдаваемые устройством при движении пера по поверхности планшета, представлены в табл.1.

Таблица 1- Значения координат и давления для буквы «г»

Время	x	y	Время	x	y
90004130	9546	2750	90004361	10430	2449
90004137	9546	2750	90004369	10307	2270
90004145	9557	2768	90004377	10168	2080
90004153	9568	2796	90004385	10022	1883
90004161	9590	2822	90004393	9880	1685
90004169	9628	2858	90004401	9744	1483
90004177	9681	2894	90004409	9617	1285
90004185	9748	2948	90004417	9504	1096
90004193	9823	3002	90004423	9407	921
90004201	9914	3060	90004431	9324	759
90004209	10011	3115	90004439	9256	615
90004217	10116	3173	90004447	9205	481
90004225	10221	3226	90004455	9171	368
90004235	10322	3276	90004463	9145	270
90004241	10420	3321	90004471	9133	184
90004249	10513	3357	90004479	9129	117
90004257	10600	3388	90004487	9148	63
90004265	10671	3411	90004495	9163	27
90004273	10731	3429	90004503	9193	4
90004281	10780	3442	90004511	9242	0
90004289	10814	3429	90004519	9309	22
90004297	10840	3411	90004527	9399	58
90004305	10851	3370	90004535	9516	121
90004313	10840	3298	90004543	9658	198

90004321	10817	3200	90004551	9820	287
90004329	10776	3078	90004559	9999	390
90004337	10716	2939	90004567	10191	499
90004345	10637	2787	90004575	10382	611
90004353	10540	2624	90004583	10573	719

На основе полученной информации можно построить классификатор, который будет базироваться на сведениях о направлении перемещения пера. Представим поверхность ввода в виде координатной плоскости. Тогда направление минимального приращения координат (в дискретном случае это расстояние между двумя соседними точками ввода) будет направлено в одну из четвертей координатной плоскости (рис. 6).

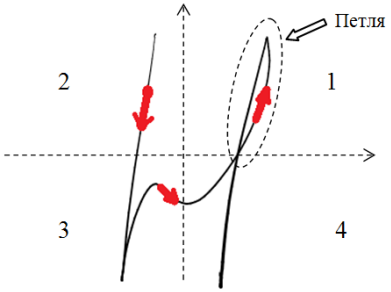

Рис. 6. Определение векторов направления на примере буквы «н»

Таким образом, вместо абсолютных координат (табл.1), можно получить набор цифр (в данном случае они не несут никакой математической нагрузки, а являются символьными обозначениями): 1, 2, 3 или 4. Они указывают направление вектора градиента для двух соседних точек. Эти направления обозначают I, II, III и IV координатные четверти соответственно. Очевидно, что если начертание символа не меняет направление, получим последовательность одинаковых чисел. Как только начертание сменит направление – цифра координатной четверти также изменится. Объединяя точки с одним направлением движения в единственное значение, получим вектор признаков, имеющий размерность гораздо меньшую, чем исходная последовательность точек ввода. Этот вектор может рассматриваться в качестве идентифицирующих данных.

Следует отметить, что не вся последовательность векторов направлений вносится в результирующий вектор. Из-за погрешности ввода или особенностей почерка возникают различные «петли» и «крючки», которые не являются частью символа (рис.6), а вводят в заблуждение

распознающий алгоритм. После замены реальных координат на номера координатных углов эти шумы остаются в виде достаточно коротких последовательностей, которые можно удалить. Опытным путем было установлено, что на качество распознавания не влияют такие последовательности, длина которых составляет менее 4,5% от общего количества точек ввода.

Анализ результатов

Результатом применения рассмотренного алгоритма является формирование кодов векторов признаков, представленных в табл. 2.

Таблица 2- Коды векторов признаков для букв русского алфавита

симво	Код	разрыв	симво	Код	разрыв	симво	Код	разрыв
а	12341341	0	л	134	0	ц	34134321	0
б	234121	0	м	3412313	0	ч	141341	0
в	1234123	0	н	31234	0	ш	34134123	0
г	14341	0	о	34123	0	щ	34134134321	0
д	12341321	0	п	314341	0	ъ	214134123	0
е	12341	0	р	314341	0	ы	3412313	1
ж	1431314134	0	с	12341	0	ь	34123	0
з	1	0	т	314313	0	э	14321	1
и	14314321	0	у	341321	0	ю	31234123	0
й	34123	1	ф	123131432	0	я	3412341341	0
к	34123121	0	х	14321341234	0			
	31341			1				

Из таблицы видно, что предложенный алгоритм в целом идентифицирует большинство символов. Совпадения кодов (например, коды букв «е» и «с») можно объяснить небрежностью ввода и особенностями почерка. Это значит, что рассмотренных признаков недостаточно, и необходимо вести дополнительные параметры, такие как среднее значение координат [5].

Список литературы

1. Бутаков Б. А. Обработка изображений на ЭВМ / Б. А. Бутаков, В. Н. Островский, И. Л. Фадеев – М.: Радио и связь, 1987. – 240 с.
2. Tapia E. Understanding Mathematics: A System for the Recognition of On-Line Handwritten Mathematical Expressions: Dissertation – Berlin, 2004. – 109 p.

3. Структурное распознавание образов: учеб. пособие / Н.М. Новикова – Воронеж: ВГУ, 2008. – 30 с.
4. Киселёв Э.Э. Система on-line распознавания рукописного текста // Информатика: проблемы, методология, технологии: Материалы XII международной научно-методической конференции. – Воронеж: ВГУ, 2012. Том 1, с. 173-175.
5. Киселёв Э.Э. Система on-line распознавания рукописного текста / Э.Э. Киселёв, С.А. Запрягаев // Сборник студенческих научных работ факультета компьютерных наук ВГУ. – Воронеж: ВГУ, 2012. С. 84.

ОЦЕНКА ПСИХОФИЗИОЛОГИЧЕСКОГО СОСТОЯНИЯ ЧЕЛОВЕКА ПО «КОМПЬЮТЕРНОМУ ПОЧЕРКУ»

студ. Р.С. Коротин,
студ. А.В. Алексеев,
доц. Я.А. Туровский

В настоящее время создание компьютерных систем оценки психофизиологического состояния человека является весьма актуальной задачей. Оценка уровня стресса пользователя при работе на компьютере может представлять значительный интерес при разработке систем принятия решений, обеспечивая необходимый уровень защиты от действий таких пользователей, которые находятся в состоянии повышенной тревоги, стресса и, следовательно, более склонны в этот период к совершению ошибок. Комплексы программ, решающих подобные задачи, существующие в настоящее время, являются крайне специализированными и не могут покрыть весь спектр имеющихся в этой области потребностей. Отсутствие простых и недорогих решений заставляет применять дорогостоящие, сложные в управлении и требующие большого количества компьютерных ресурсов системы или специальные комплексы, которые требуют затратного дополнительного обслуживания. Также пока ещё крайне затруднительно проводить анализ полученных данных и, зачастую, это приходится делать специально обученному персоналу, что увеличивает время анализа поступающей информации и, соответственно, определения психофизиологического состояния человека.

Таким образом, в настоящей работе был создан программный продукт, основанный на использовании эффекта «компьютерного

(клавиатурного) почерка». «Компьютерный почерк» – это поведенческая биометрическая характеристика, которую описывают следующие параметры [1]:

а) скорость ввода – количество введенных символов на клавиатуре компьютера в единицу времени;

б) динамика ввода, которая характеризуется интервалом времени между нажатиями клавиш и временем их удержания;

в) частота возникновения ошибок при вводе символов.

При этом временные интервалы между нажатием клавиш характеризуют темп работы, а время удержания клавиш характеризует стиль работы с клавиатурой - резкий удар или плавное нажатие. Именно анализ этих признаков лежит в основе существующих на сегодняшний день подходов изучения клавиатурного почерка [2].

Разработанный программный продукт имеет следующие особенности:

а) простота использования;

б) малая стоимость;

в) возможность работы с различными операционными системами;

г) возможность применения в маломощных системах или комплексах;

д) возможность проведения автоматического анализа состояния пользователя.

Его можно использовать в различных сферах исследований: для проведения психофизиологических исследований в виде определения психического состояния человека; для идентификации людей по «компьютерному почерку»; для проверки эффективности управления.

Для достижения поставленной цели были решены следующие задачи:

а) создание алгоритма считывания параметров нажатия клавиш в фоновом режиме, определения времени нажатия и интервала между нажатиями;

б) подсчет дисперсии, среднего времени нажатия клавиши и среднего времени между нажатиями клавиш;

в) запись полученных результатов в таблицу Microsoft Excel и их последующая обработка.

При создании программы был использован метод считывания данных о нажатых клавишах, схожий с методом «клавиатурного шпиона» [3]. Для этого разработана процедура, которая в реальном масштабе времени «отлавливает» нажатие клавиши, определяя три параметра: символ нажатой клавиши, время её нажатия и время удержания клавиши. После этого определяется средний интервал времени между нажатиями. При этом в главном окне программы появляются соответствующие символы, а в таблицу записываются параметры нажатия. Все данные обновляются при каждом новом нажатии клавиши. В главном окне программы есть две кнопки: «старт» и «стоп», а также несколько окон вывода, которые дублируют данные, записываемые в таблицу. Программа может работать в фоновом режиме.

Поскольку у каждого человека время нажатия клавиш на клавиатуре и набора слов является сугубо индивидуальным, то по ряду параметров, которые в сумме называются «компьютерным почерком», можно определять психическое состояние пользователя и выдавать сообщения об изменении такого состояния.

С помощью разработанной программы можно не только проводить психофизиологические исследования, но и осуществлять контроль эффективности управления. Его можно использовать в организациях или на предприятиях, где надо отслеживать состояние операторов в реальном масштабе времени, например, в диспетчерских службах.

Данный программный продукт обладает высоким потенциалом для широкого использования и развития. На основе этой программы могут быть созданы не только специализированные медицинские комплексы, но и системы идентификации пользователей по их «компьютерному почерку», что может потребоваться в различных жизненных ситуациях, в системах контроля при дистанционном обучении или в юридической практике.

Список литературы

1. http://lc.kubagro.ru/aidos/aidos04/1.3.htm
2. http://elib.bsu.by/bitstream/
3. http://www.staffcop.ru/articles/keylogger-klaviaturniy-shpion.php

ИНТЕГРИРОВАНИЕ МАТРИЧНЫХ АЛГЕБР ЛИ

студ. Е.Д. Кочурова,
проф. А.В. Лобода

Термины, используемые в статье

Матричная алгебра Ли - линейное пространство матриц, замкнутое относительно коммутатора $[A,B]=AB-BA$.

Под **аффинной однородностью поверхности** $M\in\mathbb{C}^3$ вблизи точки $p\in M$ подразумевается следующее её свойство: для любой близкой к p точки $q\in M$ найдутся окрестности $U(p),\ V(q)$ точек p и q, соответственно, и аффинное преобразование пространства $\mathbb{C}^3$, переводящее p в q, а $M\bigcap U(p)$ в $M\bigcap V(q)$.

Аффинное преобразование можно представить в виде:

$$\begin{pmatrix} x^* \\ y^* \\ z^* \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix},$$

Введение

В данной работе обсуждается фрагмент общего алгоритма описания аффинно-однородных вещественных гиперповерхностей 3-мерного комплексного пространства. Эта объемная задача изучается в работах Лободы А.В. (см., например, [1]). Она содержит большое количество отдельных частей, каждая из которых требует отдельного исследования.

Постановка задачи

В пространстве 3-х комплексных переменных $z_1=x_1+iy_1, z_2=x_2+iy_2, w=u+iv$ ищется функция $\Phi=-v+F(z_1,z_2,u)$, удовлетворяющая пяти уравнениям в частных производных:

$$\Re e\{E_k(\Phi)\}_{|\Phi=0}, k=1,..,5. \qquad (1)$$

Здесь каждое из 5 базисных векторных полей E_k имеет вид:

$$(A_1z_1 + A_2z_2 + A_3w + p)\frac{\partial}{\partial z_1} + (B_1z_1 + B_2z_2 + B_3w + s)\frac{\partial}{\partial z_2} +$$

$$+ (az_1 + bz_2 + cw + q)\frac{\partial}{\partial w},$$

где $A_k, B_k, a, b, c, p, q, s$ – комплексные константы.

В упрощенной форме алгебры аффинных векторных полей можно рассматривать как матричные алгебры, состоящие из матриц вида

$$\begin{bmatrix} A_1 & A_2 & A_3 & p \\ B_1 & B_2 & B_3 & s \\ a & b & c & q \\ 0 & 0 & 0 & 0 \end{bmatrix}. \qquad (2)$$

Достаточным условием существования решения системы (1) является наличие структуры алгебры Ли у вещественной линейной оболочки пяти базисных матриц (или векторных полей) вида (2). В нашей задаче эти поля имеют следующий конкретный вид (см [1]):

$$E_1 = \begin{bmatrix} -6 & 0 & -it^2 & 1 \\ -2it & 0 & 0 & 0 \\ 4i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, E_2 = \begin{bmatrix} 0 & 2t & t^2 & i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$E_3 = \begin{bmatrix} -2it & 0 & 0 & 0 \\ 0 & -4it & -it^2 & 1 \\ 0 & 4i & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, E_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2t & t^2 & i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad (3)$$

$$E_5 = \begin{bmatrix} -it^2 & 0 & 0 & 0 \\ 0 & -it^2 & 0 & 0 \\ 0 & 0 & -it^2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Требуется выписать в вещественных координатах и решить систему уравнений в частных производных. В силу необходимости большого количества технических (но точных!) вычислений при решении этой задачи используется пакет символьной математики MAPLE.

Приведение матриц к улучшенному виду

Базис (3) исходной матричной алгебры Ли устроен достаточно сложно для интегрирования этой алгебры, поэтому сначала необходимо его упростить за счет матричных подобий.

Диагонализируем подобием с матрицей

$$S = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & it^2 \end{bmatrix} \tag{4}$$

матрицу E_5, пересчитаем при таком подобии остальные базисные матрицы и получим упрощенный базис

$$E_{1s} = \begin{bmatrix} -6 & 0 & -it^2 & 0 \\ -2it & 0 & 0 & 0 \\ 4i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; E_{2s} = \begin{bmatrix} 0 & 2t & t^2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$E_{1s} = \begin{bmatrix} -6 & 0 & -it^2 & 0 \\ -2it & 0 & 0 & 0 \\ 4i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; E_{4s} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2t & t^2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5}$$

$$E_{5s} = \begin{bmatrix} -it^2 & 0 & 0 & 0 \\ 0 & -it^2 & 0 & 0 \\ 0 & 0 & -it^2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Однако матрицы E_1 и E_3 всё еще довольно сложны. Для дальнейшего упрощения вводим подобие типа $E_{1n} = C * E_{1s} * C^{-1}$ с матрицей

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{2}t & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (6)$$

Пересчитывая базис алгебры еще раз, получаем новый улучшенный базис:

$$E_{1n} = \begin{bmatrix} -6 & 0 & -it^2 & 0 \\ 0 & 0 & 0 & 0 \\ 4i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, E_{2n} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, E_{3n} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$
$$E_{4n} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, E_{5n} = \begin{bmatrix} i & 0 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \qquad (7)$$

Интегрирование алгебры

Выписываем систему (комплексных) уравнений в частных производных, описанных данным базисом и формулой (1):

$$\Re e\left\{(-6z_1 - it^2 w)\frac{\partial F}{\partial z_1} + 4iz_1 \frac{\partial F}{\partial w}\right\} = 0,$$

$$\Re e\left\{z_2 \frac{\partial F}{\partial z_1}\right\} = 0, \quad \Re e\left\{iz_2 \frac{\partial F}{\partial w}\right\} = 0, \quad \Re e\left\{z_2 \frac{\partial F}{\partial z_2}\right\} = 0, \qquad (8)$$

$$\Re e\left\{iz_1 \frac{\partial F}{\partial z_1} + iz_2 \frac{\partial F}{\partial z_2} + iw\frac{\partial F}{\partial w}\right\} = 0$$

Используя формулы для производных по комплексным переменным z_1, z_2, w:

$$\frac{\partial F}{\partial z_1}=\frac{1}{2}(\frac{\partial F}{\partial x_1}-i\frac{\partial F}{\partial y_1}),\frac{\partial F}{\partial z_2}=\frac{1}{2}(\frac{\partial F}{\partial x_2}-i\frac{\partial F}{\partial y_2}),\frac{\partial F}{\partial w}=\frac{1}{2}(\frac{\partial \Phi}{\partial u}-i\frac{\partial \Phi}{\partial v}) \qquad (9)$$

и условие v=F, отвечающее равенству Ф=0, перейдём к вещественной форме уравнений (10):

$$U_1:(-6x_1+t^2F)\frac{\partial F}{\partial x_1}-(6y_1+ut^2)\frac{\partial F}{\partial y_1}-4y_1\frac{\partial F}{\partial u}=4x_1,$$

$$U_2:x_2\frac{\partial F}{\partial x_1}-y_2\frac{\partial F}{\partial y_1}=0,$$

$$U_3:x_2\frac{\partial F}{\partial x_{21}}+y_2\frac{\partial F}{\partial y_1}=0,\quad U_4:-y_2\frac{\partial F}{\partial u}=x_2, \qquad (10)$$

$$U_5:x_1\frac{\partial F}{\partial y_1}-y_1\frac{\partial F}{\partial x_1}+x_2\frac{\partial F}{\partial y_2}-y_2\frac{\partial F}{\partial x_2}-F\frac{\partial F}{\partial u}=u.$$

В этой системе функция F зависит от переменных x_1,x_2,y_1,y_2,u. Для ее нахождения будем поочередно решать отдельные уравнения системы (10). Любое из них имеет следующий (общий) вид линейного уравнения в частных производных:

$$A_1\frac{\partial F}{\partial x_1}+A_2\frac{\partial F}{\partial x_2}+B_1\frac{\partial F}{\partial y_1}+B_2\frac{\partial F}{\partial y_2}+C\frac{\partial F}{\partial u}=D.$$

Для решения каждого такого уравнения, как известно (см. [2]), достаточно найти независимые интегралы соответствующей этому уравнению в частных производных системы ОДУ:

$$\frac{dx_1}{A_1}=\frac{dx_2}{A_2}=\frac{dy_1}{B_1}=\frac{dy_2}{B_2}=\frac{du}{C}=\frac{dF}{D}.$$

Решаем по такой схеме поочерёдно самые простые уравнения. Для U_4 выпишем независимые интегралы системы:

$$\frac{dx_1}{0}=\frac{dx_2}{0}=\frac{dy_1}{0}=\frac{dy_2}{0}=\frac{du}{-y_2}=\frac{dF}{x_2}.$$

Имеем здесь $x_1 = C_2, x_2 = C_3, y_1 = C_4, y_2 = C_5$, $\frac{du}{-y_2} = \frac{dF}{x_2}$.

Решение последнего дифференциального уравнения дает нам

$$\frac{u}{-y_2} = \frac{F}{x_2} + C_1.$$

Поэтому решение U_4 имеет вид

$$F = -\frac{x_2}{y_2}u + G(x_1, x_2, y_1, y_2).$$

Выразим производные в уравнениях через новую функцию G:

$$\frac{\partial F}{\partial x_1} = \frac{\partial G}{\partial x_1}, \frac{\partial F}{\partial x_2} = -\frac{u}{y_2} + \frac{\partial G}{\partial x_2}, \frac{\partial F}{\partial y_1} = \frac{\partial G}{\partial y_1},$$
$$\frac{\partial F}{\partial y_2} = \frac{x_2}{y_2^2}u + \frac{\partial G}{\partial y_2}, \frac{\partial F}{\partial u} = -\frac{x_2}{y_2}$$

Остальные уравнения примут вид:

$$U_1 : (-\frac{x_2 u t^2}{y_2} + Gt^2 - 6x_1)\frac{\partial G}{\partial x_1} - (6y_1 + ut^2)\frac{\partial G}{\partial y_1} = 4(\frac{y_1 x_2}{y_2} - x_1),$$
$$U_2 : x_2\frac{\partial G}{\partial x_1} + y_2\frac{\partial G}{\partial y_1} = 0, \quad U_3 : x_2\frac{\partial G}{\partial x_2} + y_2\frac{\partial G}{\partial y_2} = 0,$$
$$U_5 : x_1\frac{\partial G}{\partial y_1} - y_1\frac{\partial G}{\partial x_1} + x_2\frac{\partial G}{\partial y_2} - y_2\frac{\partial G}{\partial x_2} = -\frac{x_2}{y_2}G.$$

Далее аналогично решая U_3, получим:

$$G = H(s, x_1, y_1), \text{ где } s = \frac{y_2}{x_2}.$$

Остальные уравнения примут вид:

$$U_1 : (-\frac{ut^2}{s} - 6x_1)\frac{\partial H}{\partial x_1} - (6y_1 + ut^2)\frac{\partial H}{\partial y_1} = 4(\frac{y_1}{s} - x_1),$$

$$U_2 : x_2 \frac{\partial H}{\partial x_1} + y_2 \frac{\partial H}{\partial y_1} = 0,$$

$$U_5 : x_1 \frac{\partial H}{\partial y_1} - y_1 \frac{\partial H}{\partial x_1} + (1+s^2)\frac{\partial H}{\partial s} = -\frac{H}{s}.$$

Осталось еще одно простое уравнение U_2, решая его, получаем:

$$H = \Phi(r,s), \text{ где } r = \frac{y_1 x_2 - x_1 y_2}{x_2} = y_1 - x_1 s.$$

.

Два оставшихся сложных уравнения после всех описанных преобразований и замен примут вид:

$$\begin{aligned} U_1 &: \ (\varphi t^2 s + 6r)\frac{\partial \varphi}{\partial r} = 4\frac{r}{s}, \\ U_5 &: \ sr\frac{\partial \varphi}{\partial r} + (1+s^2)\frac{\partial \varphi}{\partial s} = -\varphi. \end{aligned} \qquad (11)$$

Для U_5 выпишем независимые интегралы соответствующей системы ОДУ:

$$\frac{dr}{rs} = \frac{ds}{(s^2+1)} - \frac{d\Phi}{-\Phi/s}.$$

Решим сначала второе из этих двух уравнений

$$\int \frac{ds}{s(s^2+1)} = \int \frac{d\varphi}{-\varphi}.$$

Упростим первый интеграл, представив его подынтегральную функцию в виде:

$$\frac{1}{s(s^2+1)} = \frac{1}{s} - \frac{s}{s^2+1}.$$

Тогда

$$\int (\frac{1}{s} - \frac{s}{s^2+1})ds = \int \frac{d\Phi}{-\Phi} \text{ или } \ln\Phi = -\ln\frac{s}{\sqrt{s^2+1}} + \ln C_1.$$

Получим:

$$\Phi = \frac{C_1\sqrt{s^2+1}}{s}.$$

Аналогично, решая первое уравнение в (11), получим:

$$C_2 = \frac{r}{\sqrt{s^2+1}}.$$

В итоге общее решение уравнения U_5 имеет вид:

$$\Phi = \frac{g(\xi)\sqrt{s^2+1}}{s}, \text{ где } \xi = \frac{r}{\sqrt{s^2+1}}.$$

Подставим это решение в первое уравнение U_1 и преобразуем его к виду:

$$(6+\frac{t^2 g}{\xi})g' = 4.$$

Это однородное ОДУ 1-го порядка приводится заменой переменной $\lambda = \frac{g}{\xi}$ к следующему уравнению с разделёнными переменными:

$$\frac{(6+t^2\lambda)d\lambda}{t^2\lambda^2+6\lambda-4} = -\frac{d\xi}{\xi}.$$

Решение последнего имеет (в неявной форме) вид:

$$(\lambda-\lambda_1)^{\alpha}\cdot(\lambda-\lambda_2)^{\beta} = \frac{C}{\xi},$$

где $\alpha + \beta = 1$, λ_1, λ_2 - два различных вещественных корня уравнения

$$\lambda^2 + \frac{6}{t^2\lambda} - \frac{4}{t^2} = 0.$$

После подстановки в предыдущее уравнение всех описанных выше замен, а также завершающей замены комплексных переменных $z_1^* = w - \lambda_1 z_1, w^* = w - \lambda_2 z_2$, получим уравнение:

$$(\Re e(z_2\overline{z}_1))^{\alpha}\cdot(\Re e(z_2\overline{w}))^{\beta} = |z_2|.$$

Отметим, что это искомое уравнение представляет новое семейство аффинно-однородных поверхностей.

Список литературы

1. Лобода А.В. Об аффинной однородности поверхностей трубчатого типа в C^3 /А.В. Лобода, Т.Т.З. Нгуен //Труды МИАН- 2012.- Т. 279.- С. 93 - 110.
2. Понтрягин Л.С. Обыкновенные дифференциальные уравнения/Л.С. Понтрягин //«Наука», 1974. - 331 с.

ИСПОЛЬЗОВАНИЕ BUSINESS STUDIO ДЛЯ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ И ФСА

студ. Е.Д. Кочурова,
студ. Е.В. Киреевская,
проф. М.Г. Матвеев

Введение

Управление компанией заключается в рациональной организации её деятельности. Анализом и оптимизацией деятельности компании занимаются топ-менеджеры или привлеченные консультанты. Их идеи и предложения по улучшению бизнеса могут быть кардинально противоположными. Понять, какая идея наиболее выигрышная – сложно. Проводить эксперименты на реальной организации - слишком дорого, в этом случае принять правильное управленческое решение, не экспериментируя над предприятием и сотрудниками, позволяет **«имитационное моделирование»** и **«функционально-стоимостной анализ» (ФСА).**

Имитационное моделирование – метод исследования, основанный на том, что изучаемая система заменяется моделью, имитирующей эту систему. Над моделью проводят эксперименты и в результате получают информацию о реальной системе. Имитационное моделирование выполнения бизнес-процессов широко применяется в проектах по реинжинирингу деятельности организации, когда необходимо заранее спрогнозировать результаты.

Показатели, которые подвергаются оптимизации в первую очередь, это:

- затраты процесса;

- продолжительность процесса;
- количество обслуженных клиентов или количество произведенного продукта.

Неудовлетворительные значения этих показателей неизбежно снижают эффективность процесса, что приводит к потере денежных средств организации и недовольству руководителя. Почему именно эти показатели выделяются как основные? Высокая стоимость процесса напрямую увеличивает затраты организации. Длительное выполнение процесса увеличивает вероятность того, что его результат будет получен не вовремя и к этому времени уже может быть никому не нужен.

Метод имитационного моделирования позволяет оценить время выполнения процесса и время, затрачиваемое на задержки в ходе выполнения процесса.

Для определения стоимости процесса совместно с имитационным моделированием целесообразно проводить функционально-стоимостной анализ (ФСА).

Функционально-стоимостной анализ – метод расчета себестоимости продукта, который отличается от традиционного тем, что фокусируется на подсчете стоимости процессов, необходимых для производства продукта или предоставления услуги, а не на разделении затрат на прямые и косвенные и группировке этих затрат по статьям калькуляции. В основе ФСА лежит положение о том, что для производства продукта необходимо выполнить ряд процессов, на выполнение которых тратятся ресурсы. Стоимость процесса рассчитывается путем переноса стоимости ресурсов на стоимость каждого шага процесса. Сумма расходов на выполнение всех шагов процессов составляет себестоимость продукта.

Проведение функционально-стоимостного анализа позволяет получить оценку себестоимости через управление процессами, направленными на производство продукта или оказание услуги. В этом состоит отличие метода функционально-стоимостного анализа от традиционных финансовых методов учета затрат, в рамках которых деятельность компании оценивается по функциональным операциям, а не по конкретным продуктам (услугам), предоставляемым заказчику. В основе функционально-стоимостного анализа лежит следующее положение: для производства продукта (услуги) необходимо выполнить ряд процессов, затратив при этом определенные ресурсы. Если традиционные методы вычисляют затраты на некоторый вид деятельности лишь по категориям расходов, то функционально-стоимостной анализ показывает стоимость

выполнения всех шагов процесса. Таким образом, методика функционально-стоимостного анализа позволяет наиболее точно определить затраты на производство продуктов (оказание услуг), а также предоставляет информацию для анализа процессов и их улучшения.

В системе Business Studio имитационное моделирование и функционально-стоимостной анализ используются параллельно. Функционально-стоимостной анализ необходим для расчета стоимости процесса. Задача имитационного моделирования – рассчитать длительность каждого шага процесса [1].

Проведение имитационного моделирования предполагает осуществление четырех основных этапов:

1. Построение модели одного или нескольких процессов, выполнение которых необходимо оптимизировать;
2. Запуск имитации выполнения процессов модели;
3. Анализ полученных показателей;
4. Повторение п.1-3 для альтернативных сценариев выполнения процесса и выбор наиболее оптимального.

Метод имитационного моделирования позволяет имитировать выполнение процесса так, как оно происходило бы в действительности, но в режиме ускоренного времени.

1. ПОСТАНОВКА ЗАДАЧИ

Целью настоящей работы является показать проведение ФСА в Business Studio на классическом примере приёма пациентов двумя врачами.

Описание задачи о врачах. Доктор А и доктор Б имеют в совместной собственности кабинет, в котором начиная с 9.00 ведут утренний приём больных. Приемная открывается в 8.30, а закрывается в 10.00 утра. Секретарь сохраняет записи об обращениях пациентов за последние десять недель, кроме того, сами врачи ведут учет пациентов, принятых ими в часы консультаций. Входной поток задаётся нормальным распределением.

Одна половина пациентов регистрируется у доктора А, другая — у доктора Б, причем они образуют две отдельные очереди, которые движутся по принципу "обслуживания в порядке прибытия"(FIFO). Однако если свободен другой доктор, то 90% пациентов высказывают желание обратиться к нему, когда подошла их очередь, а их доктор занят [2].

Распределение времени консультаций обоих докторов также описывается нормальным распределением, но с разным математическим ожиданием.

Требуется, используя имитационную модель, оценить входной поток пациентов в часы утреннего приема и ответить на следующие вопросы:

1. Какое число пациентов ожидает в приемной в 9:00 утра?
2. Чему равно среднее время ожидания пациентом приема в очереди?
3. В котором часу каждого из докторов покидает последний пациент?

2. ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ И ФСА

1. Строим модель процессов обслуживания пациентов врачами в нотации EPC (Event-Driven Process Chain – событийная цепочка процессов);
2. Для анализа количества пациентов, пришедших в часы работы приёмной, задаётся нормальный закон распределения возникновения событий.

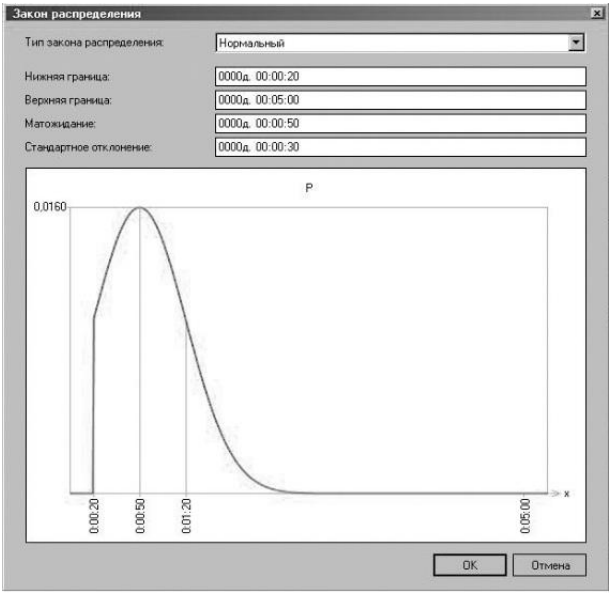

Рис. 1. Закон распределения

Business Studio позволяет также смоделировать моменты возникновения событий в течение произвольного периода.

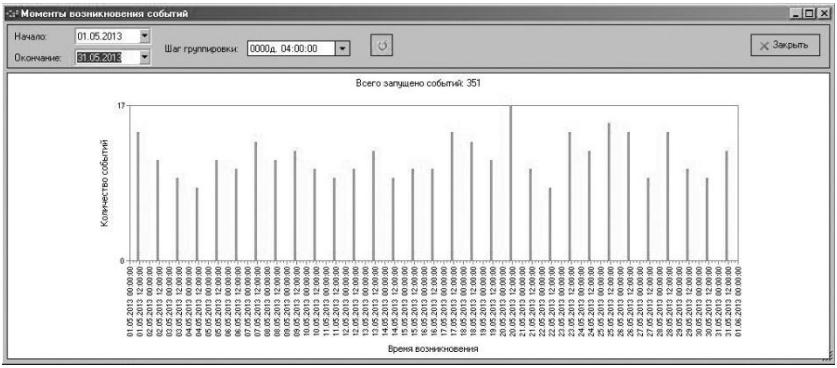

Рис. 2. Моменты времени возникновения событий

3. Для анализа времени консультации задаётся время консультации в виде нормального распределения;
4. Для проверки попадания пациентов к врачу задаётся условие перехода к соответствующим веткам процессов в виде логических выражений;

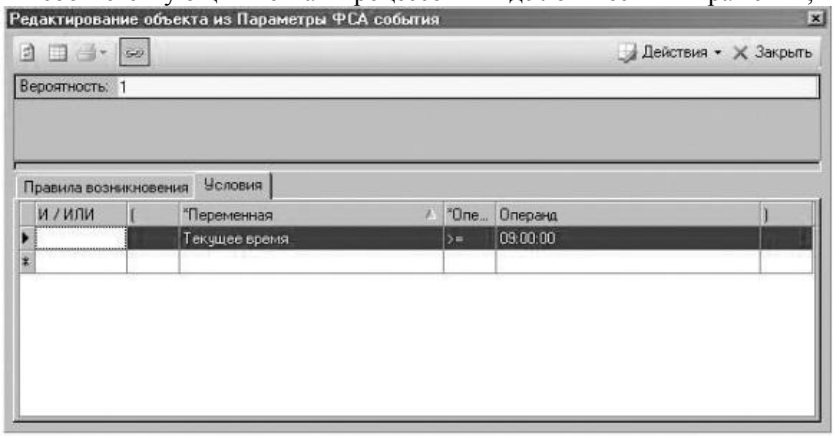

Рис. 3. Редактирование условий

5. Задаются ресурсы выполнения процессов. Для проведения консультации – врач А или Б, для проверки своего времени – пациент;
6. Задаются связи между процессами в соответствии с постановкой задачи;
7. Задаётся стоимость проведения имитации, валюта;
8. Модель бизнес процессов в нотации EPC представлена на рисунке:

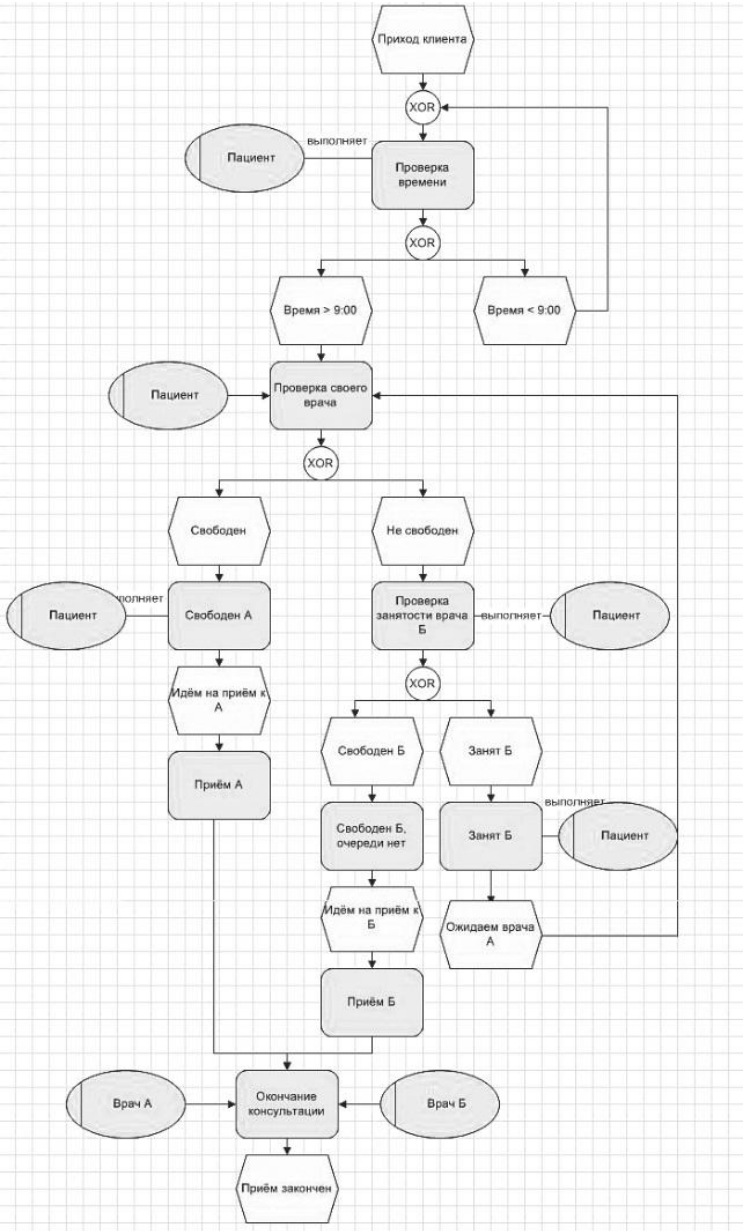

Рис. 4. Модель бизнес-процессов

9. Запускаем имитацию модели, указывая дополнительные параметры, в виде времени проведения имитации: с 8:30 до 10:00;
10. Во время имитации за выполнением каждого шага процесса можно наблюдать в пошаговом режиме, а можно запустить в авто режиме;
11. Основная информация по окончанию имитации представлена в таблице:

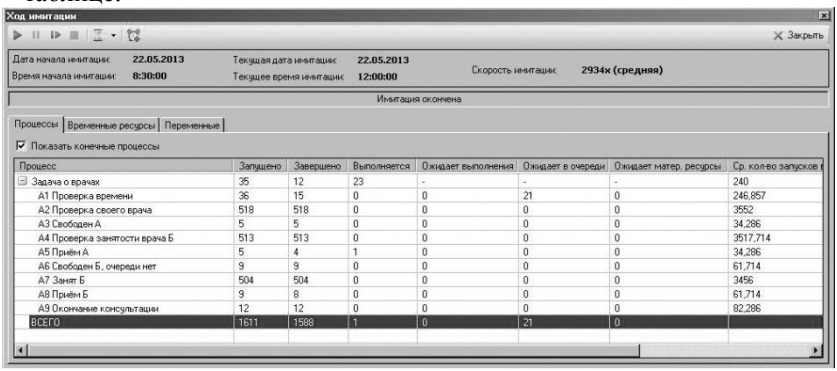

Рис. 5. Таблица имитации

12. По окончанию имитации видно:
    a. Число пациентов, ожидающих в приемной в 9:00 утра = 29;
    b. Среднее время ожидания пациентом приема в очереди можно посмотреть на гистограмме:

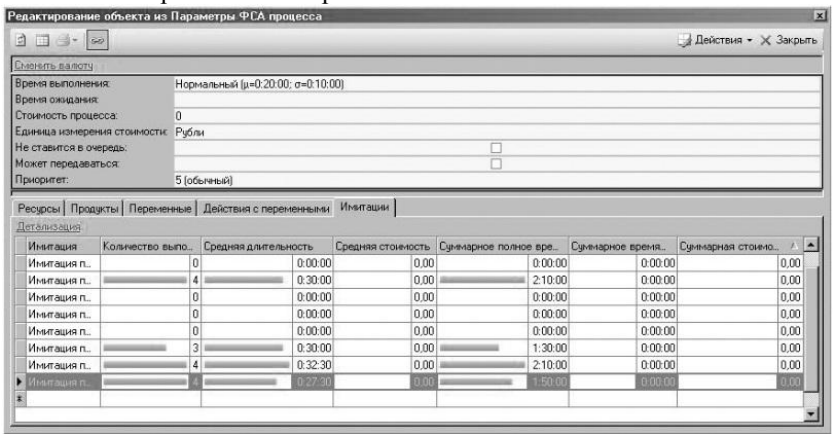

Рис.6. Гистограмма имитации бизнес-процесса

    c. Видно, что не все пациенты попали к врачу в часы приёма, в очереди осталось 21 пациент.

Выводы: на основании полученных данных в процессе имитации можно сделать вывод, что требуется увеличить часы приёма врачей, либо стоит нанять еще одного врача. Оба этих решения можно смоделировать и выяснить, что будет выгоднее.

ЗАКЛЮЧЕНИЕ

Спроектировав и настроив модель процессов, можно запускать имитацию. Поскольку временные параметры процессов и времена возникновения событий - случайные величины, один эксперимент с моделью даст только один вариант развития процесса. На основе множества повторов измерений можно получить более точные оценки показателей. Целесообразно проводить имитацию за весь период, интересующий аналитика, например, за квартал.

Таким образом, в результате проведения имитации получаются распределения значений стоимости и времени процесса, причем не только полезного времени выполнения процесса, но и времени, затраченного на ожидание необходимого количества или доступности материальных или временных ресурсов.

Если в результате анализа полученные значения показателей процесса оказались неудовлетворительными, модель можно изменить в соответствии со следующей идеей и провести имитацию снова. По результатам всех экспериментов можно выбрать вариант с наиболее оптимальными значениями показателей. При этом хочется еще раз отметить, что проведение экспериментов не останавливает работу всего предприятия и не нарушает текущее выполнение операций.

Наравне со многими плюсами метода имитационного моделирования процессов существует и ряд минусов. Нельзя не отметить тот факт, что для получения правильных результатов необходима работа по определению законов распределения случайных величин и внимательная работа по внесению всех данных для проведения имитации. Кроме того, само по себе имитационное моделирование не дает ответов на вопросы, эффективно ли работает система, являются ли значения показателей оптимальными и как перестроить бизнес-процесс. Для этого необходим бизнес-аналитик. Но только с помощью механизмов имитационного моделирования и функционально-стоимостного анализа в ряде случаев бизнес-аналитик может быстро получить и обработать ту ценную информацию, которая абсолютно необходима руководителю для принятия важных управленческих решений. Причем принять эти решения руководитель

сможет на основании сравнения значений ключевых показателей, а не поверив консультантам на слово.

Список литературы

1. Методика «Функционально-стоимостной анализ и имитационное моделирование»: [Электронный ресурс] // Система бизнес-моделирования Business Studio, 2004-2012. URL: http://www.businessstudio.ru/procedures/business/fsa_imitacia/. (Дата обращения: 31.05.2013).
2. Эддоус М. Методы принятия решений / М. Эддоус, Р.Стэнсфилд //Пер. с англ. — М.: Аудит, ЮНИТИ, 1997. - 590 с.

ПРИМЕНЕНИЕ КЛАСТЕРНОГО АНАЛИЗА И МОДЕЛЕЙ НЕЧЕТКОЙ ЛОГИКИ ДЛЯ АНАЛИЗА ЭФФЕКТИВНОСТИ БИЗНЕС-ПРОЦЕССОВ

студ. Н.Ю. Кузнецова,
доц. В.В. Гаршина

Анализ эффективности бизнес-процессов является одной из наиболее важных и в тоже время непростых задач бизнес-аналитики. Для решения этой задачи используются ключевые индикаторы производительности (KPI) [1]. KPI связывается с определенным процессом и обычно представлен численным значением. KPI может иметь нижний и/или верхний пределы, образуя диапазон или целевое значение производительности, которого должен достичь процесс. KPI можно представлять себе в виде показателя с ограничениями, составленного в свою очередь из одного или нескольких показателей. Пример простого KPI: среднее время ответа на запрос клиента меньше двух рабочих дней.

Для вычисления KPI используются различные показатели бизнес-элементов. Показатель - это измерение процесса или его элемента для оценки эффективности бизнеса. Показатель может быть автономным или использоваться в сочетании с другими показателями, что позволяет определить вычисление ключевого индикатора производительности, который измеряет производительность в отношении к бизнес-целям.

В данной статье предлагается использование методов экспертного оценивания, кластерного анализа, нечеткого вывода для выделения показателей бизнес-элементов и их значений. Из полученных значений

показателей в зависимости от бизнес-целей выбираются показатели производительности KPI.

Для определения показателей KPI предлагается использование методики, которая включает в себя последовательность решаемых задач, указанных на рисунке 1.

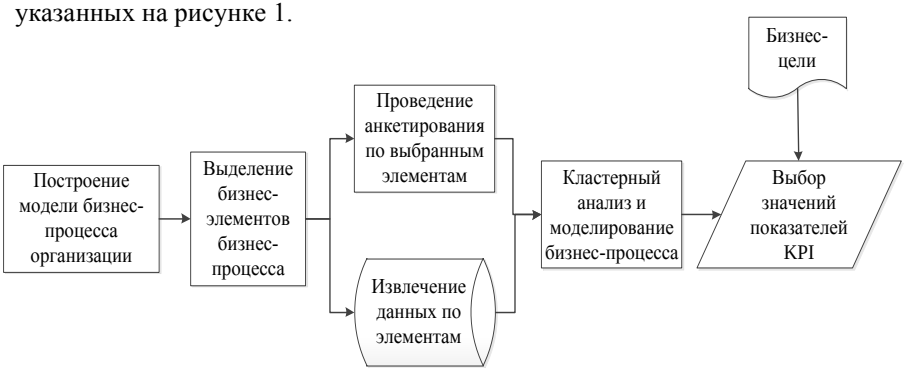

Рис. 1. Последовательность решаемых задач для определения показателей KPI

В анализируемой организации «Мегаполис», основной деятельностью которой является оказание услуг аренды транспортных средств, был выбран бизнес-процесс «Оказание услуг клиентам» (рис.2).

Соответственно, бизнес-элементами данного бизнес-процесса являются:

1) консультирование клиентов;
2) заключение договора;
3) расчеты с клиентами;
4) прием-выдача ТС;
5) тех. обслуживание;
6) урегулирование убытков;

По каждому из них было проведено анкетирование клиентов и получены данные по 50 опрошенным. Целью анкетирования являлось получение оценки по шкале 1-9 , которая отражала мнение клиента о качестве выполнения данного элемента и степени удовлетворенности клиента оказанными услугами.

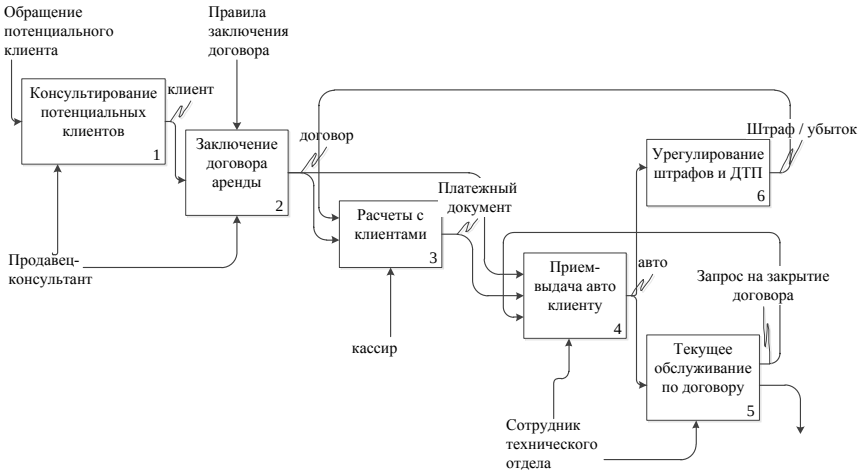

Рис. 2. IDEF0 диаграмма бизнес-процесса «оказание услуг аренды ТС» ООО «Мегаполис»

Далее, был выбран список показателей элементов и собраны данные по ним из управленческих отчетов, журналов и используемой в организации информационной системы на базе 1С-8.

Отбор показателей осуществлялся с помощью визуального анализа и анализа корреляции показателей с оценками. В результате анализа мы получили показатели для построения модели зависимости показателей элементов бизнес-процессов с оценкой, полученной по результатам опроса.

Для построения модели зависимости оценки элемента от его показателей был выбран метод горной кластеризации, также называемый методом субтрактивной кластеризации (subtractive clustering), который используется для формирования правил модели нечетких множеств (fuzzy logic). Особенностью данного метода является то, что он не требует задания количества кластеров в исследуемых данных.

Синтез нечеткой базы знаний на основе горной кластеризации осуществляется следующим способом [2]:

Обозначим через $V_1, V_2, \ldots, V_c$- центры кластеров, найденные в результате горной кластеризации. Для упрощения выкладок примем, что центры кластеров заданы двумя координатами: $V_i = (x_i, y_i), i = \overline{1,c}$. Задача состоит в синтезе нечетких правил, связывающих вход (x) с выходом (y).

Центру кластера $V_i (i = \overline{1,c})$, ставится в соответствие одно нечеткое правило:

Если $x = \tilde{x_i}$, $y = \tilde{y_i}$, в котором нечеткие термы интерпретируются как: $\tilde{x_i}$ = “ около x_i “ , $\tilde{y_i}$ = “ около y_i “. Функции принадлежностей этих нечетких термов задаются гауссовой кривой:

$$\tilde{x_i} = exp\left(-\frac{1}{2} * \left(\frac{x - x_i}{\beta}\right)^2\right), \qquad \tilde{y_i} = exp\left(-\frac{1}{2} * \left(\frac{y - y_i}{\beta}\right)^2\right),$$

$i = \overline{1,c}$,

где β – параметр, регулирующий область принадлежности.

Применение данной модели нечеткой логики выгодно отличается от остальных методов. Во-первых, позволяет формировать правила, которые могут быть использованы для анализа значений показателей и использования их для оценки формирования показателей KPI. Во-вторых, моделирование на основе центров кластеров значительно ускоряет процесс моделирования и выгодно при наличии больших объемов данных. В-третьих, моделирование на основе горной кластеризации выгодно для анализа мнения экспертов, т.к. выбирает наиболее типичное значение (центр кластера).

В Matlab горная кластеризация реализуется с помощью функции subclust, а также в составе функции genfis2 для построения модели нечеткой логики [3].

Формирование модели нечеткой логики можно продемонстрировать на примере бизнес-элемента «Урегулирование убытков».

Для создания модели нечеткой логики данные показателей «Размер убытка» (Amount_of_loss) и «Время урегулирования убытка» (Solving_time) были переданы в качестве входных переменных, данные показателя «Оценка качества» (Rating) в качестве выходной переменной. Данные были проанализированы методом горной кластеризации и были выбраны центры кластеров.

На рисунке 3 изображены значения данных (data points) и определенные алгоритмом центры кластеров (Cluster centers) для показателя «Время урегулирования, ч.» (Solving time in hours). Количество кластеров определяется алгоритмом в процессе работы, в данном случае было определено 2 кластера с центрами C1(3,5; 5) и C2(7; 4).

На основании данных центром алгоритмом были сформированы нечеткие правила, которые были использованы для построения модели нечеткой логики.

Правила, сформированные для элемента «Урегулирование убытков»:

1. if (Amount_of_loss is in1cluster1) and (Solving_time is in2cluster1) then (Rating is out1cluster1);

2. if (Amount_of_loss is in1cluster2) and (Solving_time is in2cluster2) then (Rating is out1cluster2).

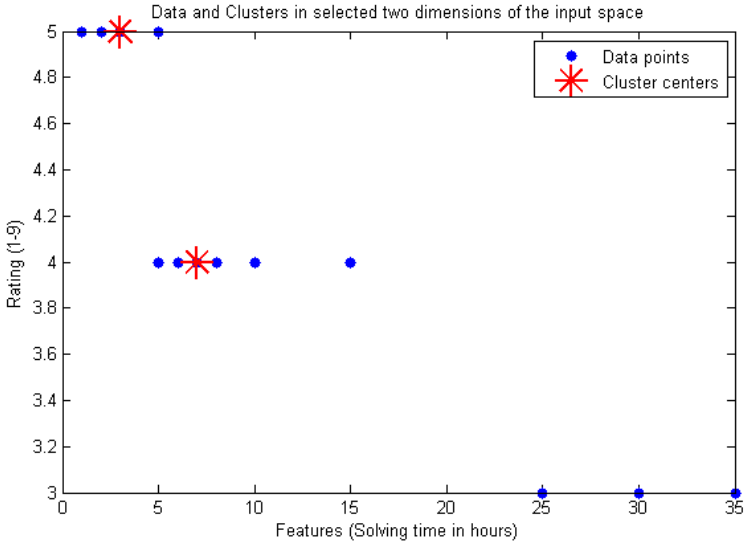

Рис. 3. Значения показателя «Решение проблемы в часах» и центры кластеров, определенные методом горной кластеризации

Как видно из правил, если значение входной переменной показателя «Размер убытка» (Amount_of_loss) принадлежит к своему первому кластеру in1cluster1, и значение второй входной переменной «Время урегулирования убытка» (Solving_time) тоже принадлежит к своему первому кластеру in2cluster1, то выходной значение должно принадлежать к первому кластеру (out1cluster1) показателя «Оценка качества» (Rating).

Аналогичное правило сформировано для кластера 2 первой входной переменной (in1cluster2) , кластера 2 второй входной переменной (in2cluster2) и кластера 2 выходной переменной (out1cluster2). Графически данные правила представлены на рисунке 4.

Как видно из рисунка 4, при убытке в 12 тыс. руб. и времени урегулирования убытка, равным 5 часам, рейтинг бизнес-элемента равен 4.42 (из 5), что является достаточно хорошим исполнением процесса.

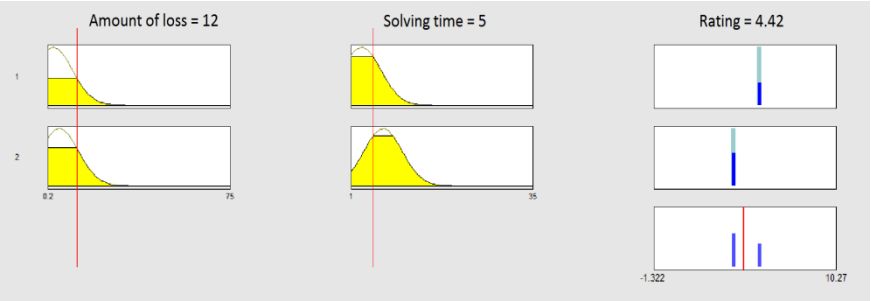

Рис. 4. Правила модели бизнес-элемента «Урегулирование убытков»

В результате для бизнес-элемента «Урегулирование убытков» была сформирована модель нечеткой логики в Matlab. На рисунке 5 представлено пространство решений, созданное с помощью полученной модели.

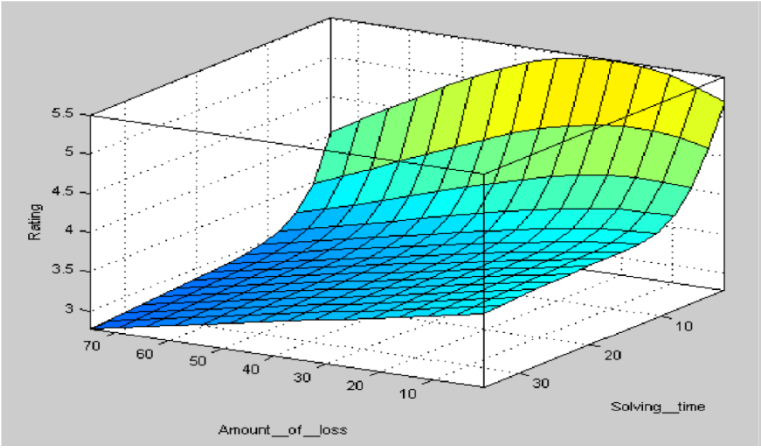

Рис. 5. Пространство рещений модели бизнес-элемента «Урегулирование убытков»

Как видно из рисунка 5, оценка качества (Rating) бизнес-процесса тем выше, чем ниже размер убытка (Amount_of_loss) и меньше время решения вопроса (Solving_time).

Модель была протестирована на тестовом множестве, результаты изображены на рисунке 6.

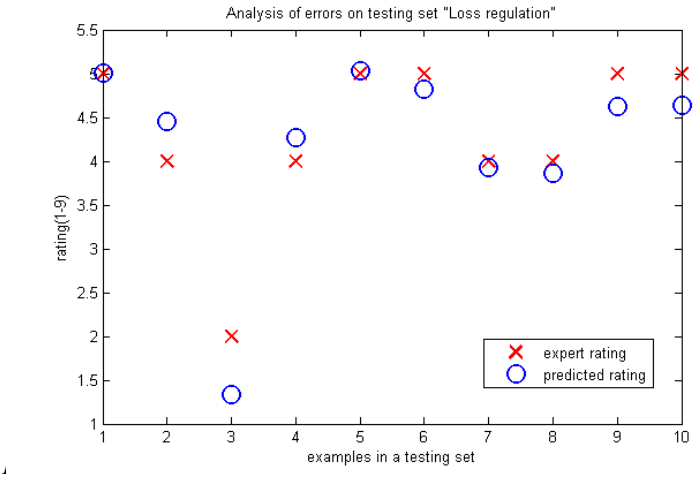

Рис. 6. Реальные значения (x) оценок элемента «Урегулирование убытков» и прогнозные (o)

При визуальном анализе прогнозных и реальных данных можно сказать, что сформированная модель с достаточной степенью точности прогнозирует оценку бизнес-элемента.

При окончании моделирования всех бизнес-элементов было проведено тестирование полученных моделей на тестовом множестве. Результаты приведены в таблице 1.

Полученные результаты показывают, что сформированные модели могут быть использованы для прогнозирования оценок качества бизнес-элементов и самого бизнес-процесса.

Таблица 1 - «Результаты тестирования полученных моделей fuzzy logic»

№	Бизнес-элемент	Ошибки, %	
		на тренировочном множестве	на тестовом множестве
1	Консультирование клиентов	10	30
2	Заключение договора	16.66	20

3	Оплата по договору	30	30
4	Выдача/прием авто в аренду	0	10
5	Техническое обслуживание	0	20
6	Урегулирование убытков	6.67	10

Таким образом, применение кластерного анализа и нечеткой логики может быть использовано для моделирования бизнес-процессов с целью формирования показателей KPI и дает неплохие результаты. Отличительные черты данного метода:

- быстрота и достаточная простота реализации;
- возможность интерпретации результатов неспециалистами в области информационных технологий;
- эффективность и точность результатов.

Описанную методику предлагается применять для оценки текущего состояния бизнес-процессов и для планирования на краткосрочную и среднесрочную перспективу. В виду естественных изменений внешних и внутренних условий функционирования организации, необходим пересмотр и переоценка показателей KPI по описанной методике с частотой, соответствующей частоте изменений.

Список литературы

1. Панов М.М. Оценка деятельности и система управления компанией на основе KPI / М.М. Панов. - М.: Инфра-М, 2012. - 255 с.
2. Ротштейн А.П. Интеллектуальные технологии идентификации: нечеткая логика, генетические алгоритмы, нейронные сети / А.П. Ротштейн. - Винница: УНИВЕРСУМ -Винница, 1999. — 320 с.
3. Леоненков А.В. Нечеткое моделирование в среде MATLAB и fuzzy TECH / А.В. Леоненков. - Санкт-Петербург: Издательство «БХВ-Петербург», 2005. – 736с.

РАЗРАБОТКА И СОЗДАНИЕ ТРЁХМЕРНОЙ СПРАВОЧНОЙ СИСТЕМЫ ЗДАНИЯ НА ПРИМЕРЕ ГЛАВНОГО КОРПУСА ВОРОНЕЖСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

студ. Д.В. Мишанин,
доц. Н.А. Тюкачев

Поиск кратчайшего пути является неотъемлемой частью жизни каждого человека. Ежедневно человеческий мозг просчитывает огромное количество вариантов достижения нужного места за наиболее короткое время и с наименьшими потерями. Иногда возникают ситуации, в которых путь до нужного здания или кабинета не известен, тогда для решения возникшей проблемы можно прибегнуть к двум вариантам: обратиться к картам или спросить, как дойти до того или иного объекта у прохожих. Поиск маршрута по карте подходит только в том случае, если пользователю нужно найди определённую улицу или здание в большом городе. Системы GPS навигации, которые сейчас встроены практически в каждое мобильное устройство, позволяют определить местоположение и проложить кратчайший маршрут до нужного места. Данная система полезна только в том случае, если вы находитесь на улице или другом открытом месте. При использовании GPS в здании, система дислоцирует местонахождение пользователя с большой погрешностью, что в свою очередь затрудняет определение начальной точки маршрута. При обращении к обычным картам, пользователь получает информацию только о местности, но поиск кратчайшего маршрута происходит своими силами. Чтобы сориентироваться в здании, можно воспользоваться пожарными планами, если на них нанесены номера аудитории. Также информацию о местоположении нужного кабинета можно спросить у стола справок, если таковой имеется.

В данной работе рассмотрена проблема навигации в здании и автоматизация поиска кратчайшего маршрута, а также отображение найденного пути на терминале или ином техническом устройстве, предназначенным для данной задачи.

В ходе выполнения работы были проанализированы с помощью метода анализа иерархии инструменты наиболее подходящие для реализации поставленной задачи, а также методы и алгоритмы нахождения наиболее оптимального маршрута.

Для успешного выполнения задача была поделена на этапы:

1. Построение трехмерной поэтажной модели главного корпуса Воронежского государственного университета, на основе планов сооружения и наполнение модели элементами для ориентирования, такими

как двери, лавочки и пр. Также на данном этапе создаются модели, которые в дальнейшем выполняют роль интерактивного меню [1,2];
2. Импортирование модели в игровой движок. Дадим определение, что такое игровой движок. Игровой движок (англ. game engine) — это центральный программный компонент компьютерных и видеоигр и других интерактивных приложений с графикой, обрабатываемой в реальном времени [3]**.** Он обеспечивает основные технологии, упрощает разработку и часто даёт игре возможность запускаться на нескольких платформах, таких как игровые консоли и настольные операционные системы, например, GNU/Linux, Mac OS X и Microsoft Windows. В данном случае с помощью этой системы становиться возможным взаимодействие между моделью и скриптами, написанными на языке высокого уровня C# [4]**.** Данные скрипты являются компонентами одной или нескольких моделей и задают поведение их, указывая, что должно произойти при взаимодействии пользователя с объектом. При импортировании моделей в движок, происходит разбиение проекта на три раздела, которые в дальнейшем именуются сценами. Для каждой из сцен предусмотрены конкретные скрипты. Первая сцена представляет собой главное меню, в котором трехмерные модели имеют функцию интерактивного интерфейса. Скрипты в первой сцены представлены на рис.1.

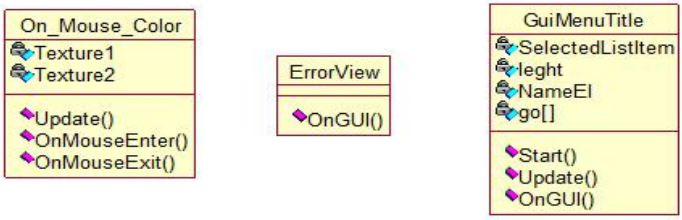

*Рис.1.*Диаграмма классов главного меню

Вторая сцена представляет собой интерфейс, который позволяет администратору заполнить базу данных, указав название аудитории или отдела, начало рабочего дня, конец рабочего дня и фамилию, имя, отчество руководителя или ответственного за данную аудиторию. Стоит отметить, что номера аудитории в базу данных заносятся автоматически, при первом запуске. В трехмерном пространстве местонахождение каждой аудитории представлено маяком с номер кабинета. Следовательно, при первом запуске программы, в базу данных заносятся имя маяка и его координаты в трёхмерном пространстве. Скрипты второй сцены представлены на рис. 2.

Рис.2. Диаграмма классов меню редактирования

Основная задача третьей сцены состоит в отображении прохода по модели от точки, где предположительно будет расположен терминал, с данным программным обеспечением, до маяка с названием выбранной аудитории. Для удобства навигации было принято решении о расположении миникарты в левом верхнем углу, на котором показана модель в изометрическом виде, а также уже пройденный маршрут. Само отображение прохода в трехмерной справочной системе происходит от первого лица;

3. В третьей сцене происходит просчёт кратчайшего маршрута. Для определения оптимального маршрута был выбран алгоритм Дейкстры. Подробное описание принципа работы алгоритма Дейкстры можно найти в [5,6]. Все маяки образуют взвешенный граф, стоимость пути которого, определяется картой проходимости. Данная карта строиться отдельно, с помощью элементов движка. Вес пути, в данном случае, равняется длине луча от точки A до точки B. В итоге составлении карты проходимости получается трёхмерный граф. На рис. 3 представлена часть графа.

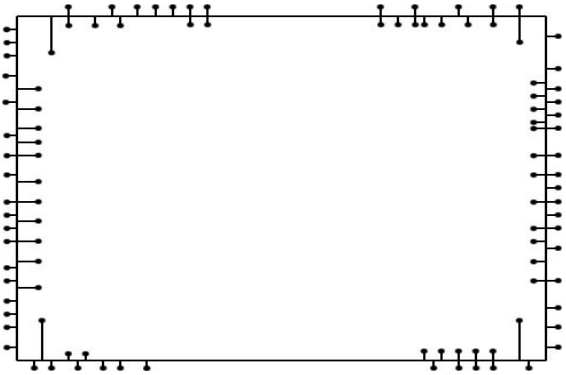

Рис.3. Часть графа трехмерной модели. Граф первого этажа здания

Скрипты второй сцены представлены на рис. 4.

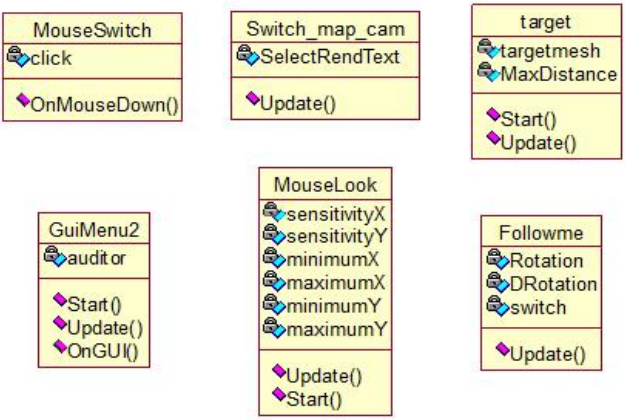

Рис.4. Диаграмма классов сцены поиска оптимального пути и прохода по заданному маршруту

При запуске программы пользователем, её выполнение происходит по следующему алгоритму:

1) пользователь выбирает с помощью выпадающего списка из базы данных номер интересующей аудитории;
2) если аудитория не выбрана, на экран выводится ошибка;
3) если аудитория выбрана, загружается сцена поиска оптимального пути и прохода по заданному маршруту;
4) происходит поиск по номеру выбранной аудитории маяка с таким же именем, и с помощью алгоритма Дейкстры, просчитывается оптимальный маршрут от предполагаемой точки терминала до точки нахождения нужного маяка;
5) перемещение модели игрока по трехмерному зданию.

Итогом работы является трехмерная справочная система здания, позволяющая в кратчайшее время просчитать оптимальный маршрут до нужной аудитории и отобразить данный маршрут на трехмерной модели. Игровой движок позволяет сделать сборку под такие платформы как Android и iOS, что делает данную справочную систему более мобильной.

Список литературы

1. Дегтярев В. Компьютерная геометрия и графика / В. Дегтярев // "Академия", 2010 г. - 192 с.
2. Шикин Е. В. Компьютерная графика / Е. В. Шикин, А.В. Боресков // Диалог-МИФИ 2005 - 464 с.
3. Энджел Э. Интерактивная компьютерная графика. Вводный курс на базе OpenGL / Э. Энджел. Вильямс, 2001 - 592 с.
4. Новиков Ф. А. Дискретная математика для программистов./ Ф. А. Новиков // Спб: Питер, 2000. - 304 с.
5. Левитин В. Глава 9. Жадные методы: Алгоритм Дейкстры // В. Левитин 2006. — С. 189—195.
6. Майника Э. Алгоритмы оптимизации на сетях и графах / Э. Майника // М.: Мир, 1981. - 324 с.

ПРЕОБРАЗОВАНИЕ ДЕКЛАРИРОВАННОГО В BPMN-НОТАЦИИ БИЗНЕС-ПРОЦЕССА В СТРЕЛОЧНЫЙ ГРАФ

студ. Д. А. Неклюдов,
проф. М. Г. Матвеев

Одной из самых распространенных спецификаций для изображения бизнес-процессов является нотация BPMN (Business Process Model and Notation) [1]. BPMN удобна своей простотой в освоении и при этой большой мощностью, позволяющей описывать любые по сложности бизнес-процессы. Это стандартизирующая спецификация призвана объединить конкурирующие стандарты описания бизнес-процессов и сделать единый подход. Модели, сделанные по спецификации BPMN, можно свободно переводить из одного графического редактора моделей в другие.

Изображение бизнес-процесса в виде стрелочного графа позволяет решать задачи календарного планирования и распределения ресурсов [2]. Но процесс преобразования бизнес-процесса не является тривиальной задачей. Для подробного рассмотрения подхода к преобразованию бизнес-процессов в стрелочный граф необходимо уточнить правила построения стрелочных графов и BPMN-аннотированных бизнес-процессов.

Таким образом, целью работы является рассмотрение основ построения bpmn-бизнес процессов и стрелочных графов и преобразование первых во вторые.

Основы стрелочных графов. В стрелочных графах каждая операция представлена стрелкой. Длина стрелок значения не имеет. Направление стрелки отражает ход времени и обычно указывается слева направо. Начало и окончание каждой операции называются событиями и изображаются на графе кружочками или узлом.

Операции обозначают буквой или словом, а события - числом. Поскольку любая операция характеризуется парой событий, ее можно также обозначать с помощью чисел, соответствующих этим событиям. Одному узлу может соответствовать (входить или выходить из него) несколько операций. Событие, изображаемое на графе с помощью узла, не считается свершившимся до тех пор, пока не окончены все входящие в него операции. Операция, выходящая из некоторого узла, не может начаться до тех пор, пока не будет достигнуто начальное событие, т.е. пока не будут завершены все операции, входящие в узловое начальное событие.

Начальным событием для С является конечное событие для А и В (рис. 1). Существенно, что в стрелочном графе сохраняется логическая зависимость операций. Иногда, чтобы достичь этого, необходимо включить в граф одну или более фиктивных логических операций.

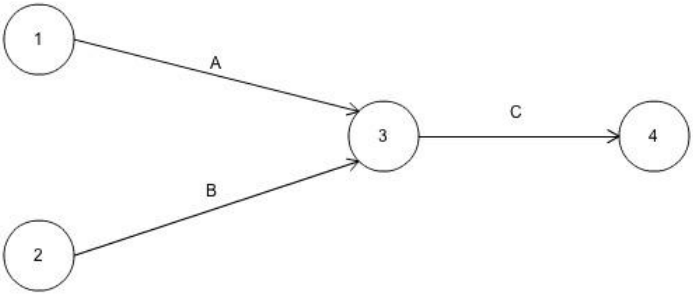

Рис. 1. Стрелочный граф

Фиктивная логическая стрелка вводится в граф, если необходимо отразить, что некоторое событие не может появиться раньше другого события, а с помощью обычных стрелок, соответствующих операциям, этого сделать нельзя. Функция фиктивной логической операции состоит в том, чтобы показать последовательность появления событий.

Фиктивным логическим операциям ставится в соответствие нулевая продолжительность выполнения, а изображаются они обычно пунктиром. Например, если работу С нельзя начать прежде, чем завершится операция

A, а работу О нельзя начать до тех пор, пока не завершатся работы А и В (рис. 2).

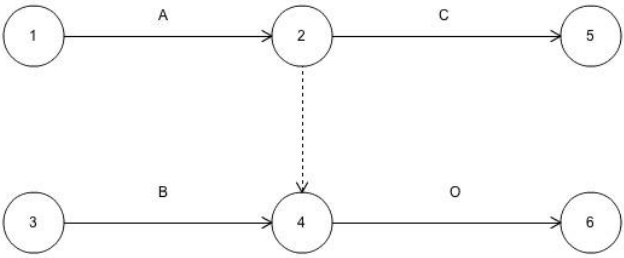

Рис. 2. Фиктивная операция

Кроме того, в стрелочных графах для избежания неоднозначности используются фиктивные операции идентификации. В некоторых пакетах прикладных программ, используемых в сетевом анализе, операции обозначаются не с помощью букв или слов, а числами, обозначающими соответствующие им события. Если же две или более операций выполняются одновременно и имеют одни и те же начальное и конечное события, то компьютер не сможет отличить их друг от друга и не воспримет вводимую исходную информацию.

Иными словами:

Стрелочным графом называется ориентированный граф (M,N), обладающий следующими свойствами:

- существует единственная вершина графа z_1, из которой дуги только выходят, т.е. не существует дуг вида (z_j, z_1);
- существует единственная вершина графа z_n, в которую дуги только входят, т.е. не существует дуг вида (z_n, z_j);
- в графе отсутствуют замкнутые пути и, следовательно, все пути в нем простые;

- пусть $z_1,\ldots,z_n$ - все вершины графа, l_j, $j=1,\ldots,m$ - все дуги графа, тогда для любой вершины z_i существует путь из z_1 в z_n, содержащий эту дугу [3].

Основы BPMN-нотации. Объекты потока управления разделяются на три основных типа: события, действия и логические операторы.

События изображаются окружностью и означают какое-либо происшествие в мире. События инициируют действия или являются их результатами.

- действия изображаются прямоугольниками со скругленными углами;

- логические операторы изображаются ромбами и представляют точки принятия решений в процессе. С помощью логических операторов организуется ветвление и синхронизация потоков управления в модели процесса;

Объекты потока управления связаны друг с другом соединяющими объектами. Существует три вида соединяющих объектов: потоки управления, потоки сообщений и ассоциации.

- поток управления изображается сплошной линией, оканчивающейся закрашенной стрелкой. Поток управления задаёт порядок выполнения действий. Если линия потока управления перечеркнута диагональной чертой со стороны узла из которого она исходит, то она обозначает поток, выполняемый по умолчанию;
- поток сообщений изображается штриховой линией, оканчивающейся открытой стрелкой. Поток сообщений показывает какими сообщениями обмениваются участники;
- ассоциации изображаются пунктирной линией, заканчивающейся стрелкой. Ассоциации используются для ассоциирования артефактов, данных или текстовых аннотаций с объектами потока управления.

На рис. 3 изображен линейный бизнес-процесс, не содержащий в себе циклов и условий.

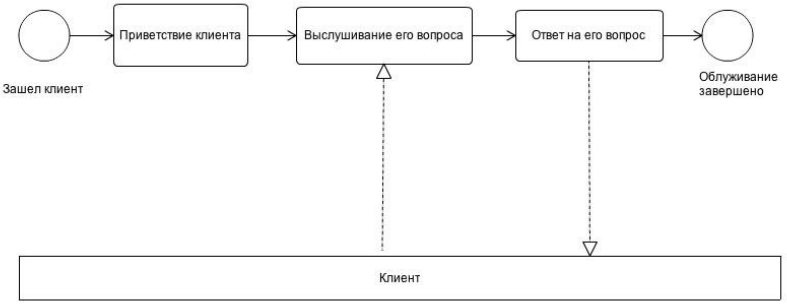

Рис. 3. Пример BPMN-аннотированного бизнес-процесса

Пример преобразования. Для проведения преобразования изображения бизнес-процесса, изображенного на рис. 3, рассмотрим процессы, изображенные на диаграмме. В процессе простейшего обслуживания клиента происходит взаимодействие между обслуживающим персоналом и клиентом. Все действия проходят

последовательно, следовательно каждое последующее событие будет зависеть лишь от одного предыдущего.

Необходимо подробнее рассмотреть взаимодействие на этапе получения ответа от пользователя. Здесь для внесения точности нужно разбить действие “выслушивание вопроса пользователя” на два: первое, это действие “Спросить, что интересует клиента”, а второе, это действие “клиент говорит, что его интересует”.

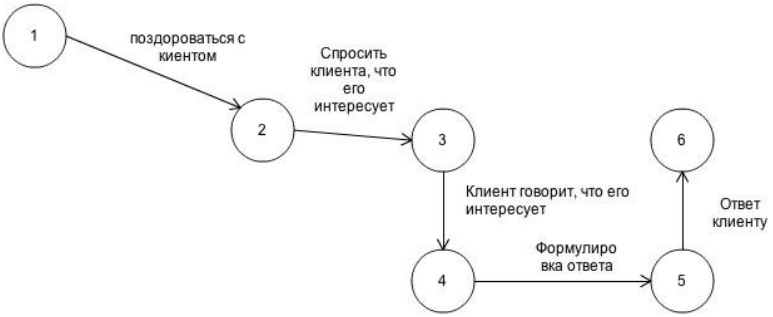

Рис. 4. Преобразованный в граф бизнес-процесс

Как видно на рис. 4, полученный граф, как и изображение бизнес-процесса, получился линейным без множественных зависимостей.

Подобное преобразование показывает процесс переведение диаграммы бизнес-процесса в стрелочный граф. На основании графа удобно решать задачи распределения ресурсов и управления запасов на складе.

Таким образом, полученные результаты показали, что представленный тип бизнес-процессов может быть успешно преобразован в стрелочный граф для дальнейшего анализа и применения методов решения различных задач.

Список литературы

1. Райн К. Л. Business Process Management (BPM) Standards / К. Л. Райн, С. Ж. Стефан, Emerald Group Publishing Limited. №15, 2009. – 48с.

2. Эддоус М. Методы принятия решений. / М. Эддоус, Р. Стенфилд // М.: Аудит. ЮНИТИ, 1997. – 590 с.

3. Косоруков О.А. Исследование операций / О.А. Косоруков, А. В. Мищенко // М.: Экзамен, 2003. – 444 с.

МОДЕЛИРОВАНИЕ СТОХАСТИЧЕСКИХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

студ. Г.С. Новицкий,
проф. М. Г. Матвеев

Важной задачей метеорологии является анализ и моделирование динамических процессов изменения атмосферной температуры. При этом рассматриваются скалярные поля температуры в трехмерном пространстве земной атмосферы. Дискретное представление этого пространства - трехмерная сетка с постоянными шагами по меридианам, параллелям и расстоянию от поверхности земли. Будем считать, что в узлах сетки производится ежедневное измерение температуры x и в каждом узле посредством случайного процесса с дискретным временем формируется временной ряд x(t).

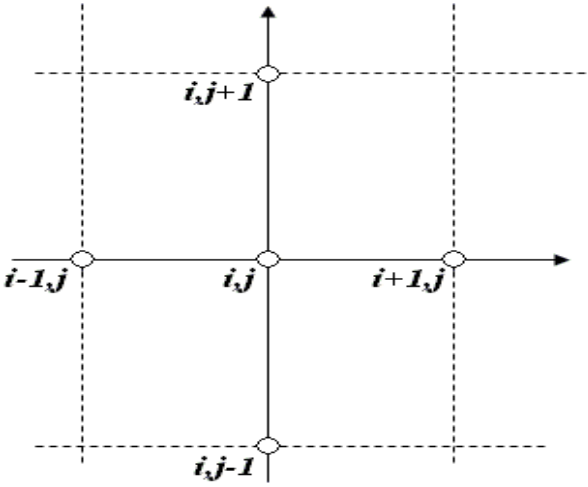

Рис.1. Фрагмент схемы измерения метеорологических показателей, шаг сетки по параллелям и меридианам составляет 2,5°

В ходе построения модели для прогнозирования дальнейших изменений температурных значений в узлах сетки x(t+1) были рассчитаны векторы градиента изменения температуры в рамках двухмерной сетки параллелей/меридиан на каждом из геопотенциальных уровней. В ходе расчетов использовалась статистика температурных значений в узлах сетки за

2001 и 2004 годы на геопотенциальном уровне равном 400.

Однако, при моделировании метеорологических процессов важно принимать во внимание тот факт, что существует множество атмосферных физических процессов, оказывающих непосредственное влияние на происходящие в атмосфере изменения. Получив данные об изменении температуры, будет правильным определить степень их зависимости от процессов переноса воздушных масс в атмосфере [1]. Имея данные о показателях ветра в каждом узле координатной сетки и полученные для каждого узла данной сетки показатели градиента температурных изменений необходимо получить величину коэффициента корреляции данных рядов.

Для каждого узла координатной сетки были вычислены следующие величины: модуль вектора градиента $|\nabla|$ и величину, определяющую направление данного вектора – косинус (синус) угла между вектором градиента и осью меридиан, а также составляющие вектора по осям.

$$|\nabla| = \sqrt{(x_{i+1,j} - x_{i-1,j})^2 + (x_{i,j+1} - x_{i,j-1})^2};$$

$$\cos\alpha = \frac{(x_{i+1,j} - x_{i-1,j})}{|\nabla|};$$

$$\sin\alpha = \frac{(x_{i,j+1} - x_{i,j-1})}{|\nabla|};$$

$$\delta = x_{i+1,j} - x_{i-1,j};$$

$$\varepsilon = x_{i,j+1} - x_{i,j-1}.$$

Здесь $x_{i,j}$ - значение температуры в узле с координатами i,j; δ - составляющая вектора по оси меридиан; ε - составляющая вектора по оси параллелей.

Теперь вычислим соответствующие показатели для вектора ветра в каждом узле используя данные u и v ветра.

$$|\nabla в| = \sqrt{u_{i,j}{}^2 + v_{i,j}{}^2};$$

$$\cos\beta = \frac{u_{i,j}}{|\nabla в|};$$

$$\sin\beta = \frac{v_{i,j}}{|\nabla в|}.$$

Здесь $u_{i,j}$ и $v_{i,j}$ - данные о ветре в узле с координатами *i,j*. $|\nabla в|$ - модуль вектора ветра.

Теперь вычислим коэффициенты корреляции полученных матриц:

Таблица 1- Значения коэффициентов корреляции

Год	2001	2004
Коэффициент корреляции по оси параллелей	0,27817	0,237226
Коэффициент корреляции по оси меридиан	0,737049	0,778219
Коэффициент корреляции косинусов	0,403402	0,421229
Коэффициент корреляции синусов	0,581707	0,679546
Коэффициент корреляции модулей	0,259609	0,27439

Как видно из полученных величин – направления векторов коррелируют заметно больше, чем их модули. В связи с этим в модели прогнозирования можно выделить больше классов по компоненту направления вектора градиента и, возможно, отказаться от классификации по его модулю, что должно привести к повышению точности прогнозирования

[2].

Также стоит отметить, что прямое влияние на коэффициент корреляции оказывают применяемые к полученным статистическим данным математические преобразования, в связи с чем вместо использования синусов и косинусов углов, для расчета которых требуются промежуточные вычисления модулей градиента изменения температуры для каждого узла сетки, следует, возможно, использовать тангенсы данных углов, для расчета которых используются только полученные составляющие вектора градиента по осям.

Обращая внимание на низкие коэффициенты корреляции полученных рядов и, в особенности, слабую корреляцию модуля градиента $|\nabla|$ и модуля ветра $|\nabla \vec{v}|$ можно рассуждать следующим образом.

При определенной среднесуточной скорости ветра в конкретном узле сетки, за определенный промежуток времени происходит перенос достаточного объема воздушных масс для изменения температуры в данном узле, соответственно, зависимость будет определяться не скоростью ветра, а температурой перенесенных воздушных масс. Ведь мощность вектора градиента характеризуется именно величиной изменения температуры Δt, на что величина скорости ветра прямого влияния не имеет, требуется только, чтобы она была достаточна для переноса необходимого объема воздушных масс [1].

Стоит обратить внимание на тот факт, что наша сетка является условно трехмерной: геопотенциальный уровень не является плоскостью на определенной высоте – это поверхность, высота которой в разных точках может ощутимо изменяться, в связи с чем может возникнуть заметная погрешность расчетов, тогда, возможно, следует построить условно трехмерную сетку и использовать в расчетах данные еще и в узлах сетки на соседних геопотенциальных уровнях.

В построении модели прогнозирования изменений атмосферной температуры необходимо выбрать верный критерий классификации точек – узлов условно трехмерной сетки. В ходе данной работы было выявлено, что большее значение в прогнозировании моделируемого процесса имеет направление градиента изменения температур, так как была выявлена его зависимость от направления ветра в данных узлах сетки. Были также выявлены погрешности, понижающие точность получаемых значений, избавившись от которых, можно повысить эффективность не только классификации статистических данных, но и самой модели прогнозирования.

Список литературы

1. Матвеев Л.Т. Курс общей физики атмосферы / Л.Т. Матвеев //2-е изд. Л.: Гидрометеоиздат, 1984. – 687 с.
2. Использование модели Сугено для прогнозирования метеорологических показателей. / М.Г. Матвеев, В.В. Михайлов, М.Е. Семенов // Вестник ВГУ, серия «Системный анализ и информационные технологии», научный журнал, №2, 2011.- стр. 164-169.

КОМПЬЮТЕРНЫЕ АЛГОРИТМЫ РЕШЕНИЯ НЕКОТОРЫХ СИСТЕМ КВАДРАТИЧНЫХ УРАВНЕНИЙ

студ. Е.А. Павлова,
проф. А.В. Лобода

1. Формулировка основного результата

В данной работе рассматривается система уравнений относительно 16 неизвестных вещественных параметров:

$$t_1, t_3, t_4, t_7, t_8, t_9; t_{15}, t_{16}, t_{17}, t_{24}, t_{25}, t_{32}; m_1, m_2, m_3, m_4 .$$

Эта система состоит из 11 квадратичных комплексных уравнений:

$$\begin{aligned} Z1:\ & -t_{17}t_{24} + 2it_9 - 22it_{25}m_4 + 14im_4^2 + 16t_{17}t_{25} - 17t_{17}m_4 + \\ & +10t_{17}t_4 - 11t_{25}m_3 + 12m_3m_4 - 8t_4m_3 - 3it_3t_{32} + 4t_7t_9 - 11it_4m_4 - \\ & -6it_{15}t_9 - 6t_{25}t_3 + 6t_3m_4 - 4t_4t_3 + 25it_3m_3 + 6it_{17}t_{32} + 2t_1t_{15} - \\ & -it_{24}t_4 - 2it_{24}t_{25} + 2it_{24}m_4 + 26im_3^2 - 6it_{32}m_3 - 24it_{17}t_3 - 2t_1t_9 + \\ & +6it_3^2 - 50it_{17}m_3 - 2t_9m_1 - 4t_7t_{15} + 4t_{15}m_1 + 4it_{15}^2 + 3t_{25}t_{32} - \\ & -4m_4t_{32} + 8it_{25}^2 + 2it_4^2 + 24it_{17}^2 + 8it_4t_{25} + 2t_4t_{32} = 0, \end{aligned}$$

$$Z2:\ 4t_3t_{15}-6m_2t_3-10im_4t_{15}-2im_2t_4+2it_3t_{16}+6im_3t_{16}-$$
$$-4it_{17}t_{16}+10m_2t_{17}-6t_{17}t_{15}-12m_3m_2+8m_3t_{15}+16it_{25}t_{15}-$$
$$-2t_{24}t_{16}-2t_{25}t_{16}-6im_2t_{25}-2t_{32}t_{15}+4m_2t_{32}+6i_4t_{15}=0,$$

$$Z3:\ m_4t_4+8m_4t_{25}+m_3t_3-8m_3t_{17}+t_{32}t_3+it_{24}t_3-im_4t_3-$$
$$-8im_4m_3-4it_4t_3+im_3t_4+8im_3t_{25}+8im_4t_{17}+it_4t_{32}-$$
$$-8it_{17}t_{25}-t_4t_{24}+4t_{17}^2+4m_3^2-2t_3^2-4t_{25}^2-4m_4^2+2t_4^2=0,$$

$$Z4:\ 2t_{15}t_{32}-im_3t_7+4m_3t_{15}+it_4t_8-4im_4t_{15}-im_4t_8+$$
$$it_{17}t_7-2t_{16}t_{25}-2m_3m_2-2t_{32}m_2-4t_3t_{15}+4t_3m_2-2t_{17}t_{15}-$$
$$it_{32}t_7-2it_{24}m_2+it_8t_{25}-it_3t_7-4it_{15}t_4-2it_{17}t_{16}+it_8t_{24}+$$
$$+2it_{15}t_{24}+2im_3t_{16}+2im_4m_2+4it_4m_2+2it_{15}t_{25}-t_{17}t_8+$$
$$+m_3t_8+t_3t_8+t_{24}t_7+t_{32}t_8+t_{25}t_7-m_4t_7+t_4t_7+2m_4t_{16}=0,$$

$$Z5:\ -2it_{16}m_3+3t_9t_{17}-2t_9m_3+6it_{25}t_{15}-6it_{15}m_4+2t_{17}m_2-$$
$$4m_2m_3-2m_2t_3-4im_2t_4+8im_2m_4-t_9t_3-8im_2t_{25}+2it_{16}t_{17}-$$
$$-t_{16}t_{25}-it_{16}t_3++3it_{15}t_4-3t_{15}t_{17}+4t_{15}m_3+2t_{15}t_3=0,$$

$$Z6:\ -i(-3m_3+3im_4+t_{32}-2t_3+4t_{17}-4it_{25}-2it_4+it_{24})\cdot$$
$$(-2m_3+2im_4-t_3+2t_{17}-2it_{25}-it_4)=0,$$

$$Z7:\ -2i(2m_2-2t_{15}+t_9)(m_2-t_{15})=0,$$

$$Z8: \quad -5t_3t_{15} + 4m_2t_3 + it_{24}t_{15} + 9im_4t_{15} - it_3t_{16} + 4im_2t_4 -$$
$$-8m_2t_{17} + 11t_{15}t_{17} + 6m_2m_3 - 9m_3t_{15} + t_4t_{16} - it_{32}t_{16} + 3t_{25}t_{16} -$$
$$-3im_3t_{16} - 3m_4t_{16} - 2im_2t_{24} + t_{15}t_{32} - 2m_2t_{32} + 3it_{16}t_{17} -$$
$$-11it_{15}t_{25} + 8im_2t_{25} - 5it_4t_{15} - 6im_2m_4 = 0,$$

$$Z9: \quad 2t_{16}t_{25} - 8m_3m_2 + 8t_{17}m_2 - 16t_{17}t_{25} - 4t_3m_2 - 2t_{16}m_4 +$$
$$+2it_{15}t_4 - it_9t_4 + 3t_{15}t_3 + t_{16}t_4 - 4it_{15}m_4 - it_{16}t_{17} + 2it_9m_4 -$$
$$-2it_{25}m_2 + 4im_4m_2 - 3it_9t_{25} + 3it_{25}t_{15} + 6m_3t_{15} - 2im_2t_4 = 0,$$

$$Z10: \quad -4im_3t_{15} + 4it_{17}t_{15} - 2t_3t_{16} + it_1t_{25} + it_1t_4 + 10it_3t_{15} -$$
$$-6it_3m_2 - 2it_6t_4 + 10it_9m_3 + 6im_3t_8 + 3it_{25}t_7 - 5it_8t_{17} +$$
$$+3it_8t_3 + 2it_{25}m_1 + 10t_9m_4 - 8t_9t_{25} + 6t_8m_4 - 5t_8t_{25} + 6t_7m_3 -$$
$$-3t_7t_{17} + 6t_4m_2 - 10t_{15}t_4 + it_7t_4 + t_7t_3 + 4t_{15}t_{25} - 4t_{15}m_4 -$$
$$-2t_{17}m_1 + 2t_3m_1 - 3t_4t_8 + t_3t_1 - t_{17}t_1 + 2t_{25}m_2 - 4m_4m_2 -$$
$$-4im_3m_2 - 8it_{17}t_9 + 2t_9t_4 + 2it_4m_1 + 2it_{17}m_2 - 2it_3t_9 - 6it_7m_4 = 0,$$

$$Z11: \quad t_{24}t_3 + 3m_3t_{24} - 18it_{17}m_3 + 9im_3t_3 + 7im_4t_{24} - 6it_{24}t_{25} -$$
$$-3it_{24}t_4 - 4it_{15}t_9 - 9im_4t_4 + 4im_2t_9 - 12im_2t_{15} - 18im_4t_{25} -$$
$$-2t_{24}t_{17} + 4im_2^2 + 8it_{15}^2 + it_{16}t_1 + it_9t_8 + 10im_3^2 + 10im_4^2 + 2t_{32}t_3 -$$
$$-2m_3t_{25} - m_3t_4 - 3it_{32}t_3 + t_7t_9 - m_4t_3 - 2m_2t_1 - 4m_2m_1$$
$$+6it_{32}t_{17} + 4m_2t_7 + 3t_1t_{15} + it_{24}^2 + it_{32}^2 - 8t_7t_{15} + 6m_1t_{15} -$$
$$-7it_{32}m_3 - 2it_{16}t_7 + 2it_{16}m_1 - 2it_8t_{15} - 2t_{32}t_{25} + 3t_{32}m_4 + 8it_{25}^2 +$$
$$+2it_4^2 + 2it_3^2 + 8it_{17}^2 + 8it_{25}t_4 - t_{32}t_4 - 8it_{17}t_3 = 0.$$

Системы, подобные данной, возникают в связи с задачей описания однородных подмногообразий вещественных и комплексных пространств (см [1]).

Решением системы, как обычно, называется любой набор параметров, обращающий все уравнения системы в верные тождества. Отметим, что некоторую информацию о возможных решениях системы легко получить из самых простых ее уравнений. Однако в задаче об однородности требуется найти все решения данной системы. В таком случае от необходимости практически полного перебора всех уравнений системы не освобождает даже относительно простая структура некоторых из них.

Основная цель работы - нахождение всех решений системы на основе разработанного компьютерного алгоритма. Такие решения являются начальными данными для дальнейшего построения матричных алгебр Ли, их последующего интегрирования и получения искомых однородных поверхностей.
Вся процедура решения распадается на большое количество случаев, в каждом из которых остается 2-5 вещественных параметров. Основной результат работы представлен ниже в виде следующей теоремы, имеющей чисто "математическую" формулировку.

Теорема. Множество всех решений системы, полученных по нашему алгоритму, охватывается следующим списком из 13 наборов параметров[1]:

1. (3) $-2m_1, \frac{1}{2}t_{32}, \frac{1}{2}t_{24}, 0,0,0;0,0,0, t_{24}, 0, t_{32};0,0,0, m_1;$
2. (30) $-2m_1, 0,0,0,0,2t_{15}; t_{15}, t_{16}, 0,0,0,0;0,0,0, m_1;$
3. (64)
   $t_1, -\frac{1}{2}t_{32}, -\frac{1}{2}t_{24}, 0,0,0;0,0, -\frac{1}{2}t_{32}, t_{24}, \frac{1}{2}t_{24}, t_{32};0,0,0, m_1;$
4. (98) $4t_7 - 2m_1, 0,0, t_7, 0,0;0,0,0,0,0,0,0;0,0,0, m_1;$
5. (100)
   $-2m_1, 2m_3, -2m_4, 0,0,0;0,0,2m_3, -m_4, 2m_4, m_3; m_4, m_3, 0, m_1;$

[1] В скобках указан порядковый номер решения, присвоенный ему алгоритмом

6. (101)

$$-2m_1, m_3, -m_4, 0,0,0;0,0, \frac{3}{2}m_3, 0, \frac{3}{2}m_4, 0; m_4, m_3, 0, m_1;$$

7. (117) $-2m_1, 0,0,0, -2t_{15}, 2t_{15}; t_{15}, t_{16}, 0,0,0,0;0,0,0, m_1;$

8. (118) $2t_7 - 2m_1, 0,0, t_7, t_8, 2t_{15}; t_{15}, t_{16}, 0,0,0,0;0,0,0, m_1;$

9. (120) $\frac{5}{2}t_7 - 2m_1, 0,0, t_7, t_8, 2m_2; 2m_2, 0,0,0,0,0;0,0, m_2, m_1;$

10. (125) $-2m_1, 0,0,0,0,0;0,0, m_3, -m_4, m_4, m_3; m_4, m_3, 0, m_1;$

11. (142) $-2m_1, 0,0,0,0, m_2; m_2, t_{16}, 0,0,0,0;0,0, m_2, m_1;$

12. (147) $2t_7 - 2m_1, 0,0, t_7, t_8, 2m_2; m_2, t_{16}, 0,0,0,0;0,0, m_2, m_1;$

13. (154)

$$3t_7 - 2m_1, 0,0, t_7, 0, m_2; m_2, 0, m_3, 0, m_4, 0; m_4, m_3, m_2, m_1;$$

Комментарии к полученным решениям:

1. В результате работы данного алгоритма было получено 156 наборов параметров;
2. 17 ситуаций, связанных с этими наборами, не доводятся до получения решений при использовании данного алгоритма. Причем в 12 из них остается по одному квадратичному неприводимому уравнению, в 3 ситуациях – по два, а оставшиеся две ситуации оказываются почти полностью нерешенными.

Пример (случай 119). Осталось одно неприводимое квадратичное уравнение

Z11: $2it_9^2 + 4t_9m_1 + 2t_1t_9 + it_{24}^2 + it_{32}^2 = 0.$

Набор переменных, которые алгоритм выразил из системы:

$$t_1, 0,0,0, -t_9, t_9; t_9, 0,0, t_{24}, 0, t_{32}; 0,0, \frac{1}{2}t_9, m_1;$$

Пример (случай 114). Осталось 7 нерешенных уравнений

Z1: $2it_9^2 + 4it_{15}^2 + 2t_{15}t_{17} - 2t_9t_7 - m_3t_{24} - 6it_9t_{15} - t_{32}m_4 = 0,$

$Z2:\ 2it_{16}m_3 + t_9m_3 + 3im_4t_9 - 2t_{16}t_{24} - 2t_{16}m_4 + 2t_{15}t_{32} - 2t_9t_{32} = 0,$

Z3: $0=0,$

$Z4:\ -it_9m_4-2it_9t_{24}+2it_{15}t_{24}+t_9m_3-2t_9t_{32}-it_7t_{32}+t_7t_{24}=0,$

Z5: $2t_9m_9-m_3t_{15}-t_{16}m_4=0,$

Z6: 0=0,

Z7:

$Z7: it_9t_{24}-it_{15}t_{24}+t_9m_3+t_{16}t_{24}-it_{16}t_{32}-t_{15}t_{32}+t_9t_{32}-it_9m_4=0,$

Z8: $-it_{16}m_3+im_4t_{15}-2im_4t_9=0,$

Z9: $0=0,$

Z10: $0=0,$

$Z11:\ it_{32}^2-4it_9^2+2t_9t_7-im_3t_{32}+10it_9t_{15}+it_7t_{16}+im_4t_{24}+m_3t_{24}-$

$-4it_{15}^2-t_{15}t_7+it_{24}^2+t_{32}m_4=0,$

Набор переменных:

$3t_7-2m_1,0,0,t_7,-3t_9+2t_{15},t_9;t_{15},t_{16},m_3,t_{24},m_4,t_{32};$

$m_4,m_3,-\frac{1}{2}t_9+t_{15},m_1;$

3. Среди полученных наборов, являющихся решениями системы, в 51 ситуации эти решения - нулевые; также выявлено 56 попарно совпадающих ненулевых наборов, являющихся решениями основной системы;
4. Выявлено 19 пар наборов, в которых один из наборов пары является частным случаем другого.

Пример. Решение 61:

$-2m_1,-\frac{1}{2}t_{32},-\frac{1}{2}t_{24},0,0,0;0,0,-\frac{1}{2}t_{32},t_{24},\frac{1}{2}t_{24},t_{32};0,0,0,m_1;$

Решение 64:

$t_1,-\frac{1}{2}t_{32},-\frac{1}{2}t_{24},0,0,0;0,0,-\frac{1}{2}t_{32},t_{24},\frac{1}{2}t_{24},t_{32};0,0,0,m_1;$

Здесь Решение 61 является частным случаем Решения 64 при t[1] = -2 m[1].

2. Описание алгоритма решения системы

Для компьютерного решения данной задачи, потребовалось 6 процедур, часть из которых имеют довольно простую структуру и несут вспомогательный характер (например, определение длины списка), а часть устроены более сложно и играют ключевую роль в данном алгоритме. Ниже приведен список этих процедур[2]:

1. Основные процедуры:
   - *factorization* - процедура разложения уравнений на множители. На вход подается список из левых частей уравнений. Далее находится первое уравнение, которое можно разложить на множители, после чего оно разлагается и процедура завершается. На выходе получаем список из множителей.
   - *complexEqu* – процедура отделения вещественных и мнимых частей. На вход подается левая часть уравнения. На выходе получается список, состоящий из мнимой и действительной части.
   - *expressVariable* - процедура выражения переменной. На вход подается левая часть уравнения, на выходе получается выраженная переменная.
2. Вспомогательные процедуры:
   - *lengthList* – процедура, результатом которой является целое число – длина списка.
   - *subsInSolve* – процедура подстановки. На вход подается список выраженных переменных. На выходе получаем тот же список, но уже с учетом подстановок.
   - *sravnenie* – процедура, которая сравнивает 2 списка. Если они равны выдает false, а если различны – true.

После описания процедур начинается сам алгоритм решения системы.

1. Составляем список, содержащий все 11 уравнений.
2. Ищем в этом списке первое уравнение, левая часть которого раскладывается на множители (делается это с помощью процедуры factorization).

Пример. В силу уравнения Z7 для любого решения нашей системы равны нулю либо первая скобка $(2m_2 - 2t_{15} + t_9)$, либо вторая $(m_2 - t_{15})$.

[2] Для разработки данного алгоритма использовался пакет символьных вычислений MAPLE (и, в частности, такие стандартные процедуры как: factor, irreduc, op и др.)

При этом подстановка, ($t_9 = 2t_{15} - 2m_2$) или $t_{15} = m_2$ в оставшиеся уравнения нашей системы приводит к появлению среди них новых приводимых уравнений. В итоге можно разложить на множители очередное уравнение и т.д.

3. Из очередного множителя, выделяем вещественную и мнимую части (процедура complexEqu), выражаем переменную (expressVariable) и заносим в список решений.

4. Далее происходит проверка на существование уравнений, которые можно разложить на множители с учетом уже проделанных действий.

- Если такие существуют, то действия повторяются, начиная со 2-ого пункта.
- Иначе, если приводимых уравнений больше нет, найденные переменные записываются в итоговый список решений и происходит переход к следующему множителю (пункт 3).

После проведения всех процедур мы получаем большое количество решений и несколько меньшее количество (а именно, 17) "остатков".

Определение. *Остатком* будем называть результат работы алгоритма, не являющийся решением.

Таким образом, с помощью данного алгоритма часть ситуаций, а именно 17, оказываются не доведенными до конца. Собственно решений получается 139 (включая нулевые и совпадающие решения).

Следующим шагом является выделение в общем списке решений нулевых и совпадающих случаев (используется процедура sravnenie).

В итоге, после корректировки всего множества результатов получаем 13 типов решений.

3. Пример семейства матричных алгебр, полученных на основе решенной системы

Найденные таким образом решения основной системы являются начальными данными для дальнейшего построения матричных алгебр Ли, их последующего интегрирования и получения искомых однородных поверхностей. Например, решению 120 соответствует семейство матричных алгебр Ли с базисами вида

$$E_1 = \begin{pmatrix} \alpha & 0 & 0 & 1 \\ 0 & 2\alpha & 0 & 0 \\ 4i & 0 & 4\alpha & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_2 = \begin{pmatrix} 0 & 0 & 0 & i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 2i & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$E_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & i \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Интегрирование алгебр приводит к новому семейству аффинно-однородных алгебраических поверхностей 4-ого порядка.

$$v = \left| z_2 \right|^2 \pm x_1^4 + Ax_1^3 + Bx_1^2$$

Здесь коэффициенты A и B выражаются некоторым образом через параметр α исходной алгебры.

Список литературы

1. Лобода А.В., Об аффинной однородности поверхностей трубчатого типа в C^3 /А.В.Лобода, Т.Т.З. Нгуен //Труды МИАН - 2012.- Т. 279.- С. 93 - 110.

2. Кокс Д. Идеалы, многообразия и алгоритмы / Д.Кокс, Дж. Литтл, Д.О'Ши // М.: Мир, 2000.- 688с.

ИНТЕРНЕТ-САЙТ СТОМАТОЛОГИЧЕСКОЙ ПОЛИКЛИНИКИ С АКЦЕНТОМ НА ПРОФИЛАКТИКУ ЗАБОЛЕВАНИЙ

студ. Д.И. Саранцев, студ. А.А. Смолина (ВГМА), студ. Е.В. Калмыкова (ВГМА), студ. В.Р. Малинин (ВГМА), студ. Д.Э. Мун (ВГМА),

проф. С.Д. Кургалин, асс. С.В.Борзунов, доц. Л.П. Друганова (ВГМА), проф. В.А. Кунин (ВГМА), проф. А.В. Сущенко (ВГМА)

Стоматологическая поликлиника Воронежской государственной медицинской академии им. Н.Н.Бурденко (ВГМА) является одним из ведущих лечебных учреждений стоматологического профиля в регионе, а также учебной базой для студентов стоматологического факультета, клинических интернов, ординаторов и аспирантов. На базе поликлиники работают семь кафедр стоматологического профиля: клинической и детской стоматологии, ортопедической стоматологии, поликлинической и пропедевтической стоматологии, терапевтической стоматологии, хирургической стоматологии с челюстно-лицевой хирургией, ортопедической стоматологии, стоматологии общей практики. Поликлиника включена в программу модернизации здравоохранения, в рамках которой проводится комплексная работа, направленная на улучшение качества оказания стоматологических услуг в терапевтической, хирургической, ортопедической, детской стоматологии и ортодонтии. В клинике широко используются инновационные методы диагностики и лечения заболеваний, она сертифицирована по 25 видам медицинской деятельности и имеет сертификат соответствия и высшую категорию лечебного учреждения, где работают специалисты, имеющие личные лечебные методики, защищенные патентами. Здесь можно провести лечение и профилактику кариеса зубов и его осложнений, болезней пародонта, получить качественную стоматологическую помощь при лицевых болях, при болезнях слизистой оболочки полости рта и губ и др. Пациентов обследуют и при необходимости прооперируют с помощью эндоскопического оборудования при заболеваниях височно-нижнечелюстного сустава, при болезнях слюнных желез, а также произведут установку имплантата при утрате зубов для восстановления функции жевания. При выраженных деформациях зубочелюстной системы

может быть изготовлен сложно-челюстной протез, а также проведено эстетическое протезирование.

Профилактика стоматологических заболеваний – это вопрос, постоянно беспокоящий стоматологов. По данным Всемирной организации здравоохранения и ряда авторов, распространенность стоматологических заболеваний среди детского населения достигает 75-95%, взрослого – 100%.

Учитывая высокую распространенность стоматологических заболеваний и тенденцию к ее росту, был создан сайт, целью которого является информирование населения о современных методах и средствах профилактики заболеваний полости рта, методах их лечения, о новинках литературы, а также представление информации, относящейся непосредственно к деятельности поликлиники ВГМА, создание условий для активного общения персонала поликлиники и пациентов и др.

Среди задач, реализованных в ходе работ по созданию многофункционального сайта поликлиники, можно выделить следующие: построение на базе информационных технологий полноценного веб-проекта; анализ эффективности работы интернет-сайта с целью максимального привлечения внимания пользователей; модернизация и развитие созданного сайта; разработка механизма обратной связи администратора сайта с его посетителями.

На сайте реализованы: регистрация и аутентификация пользователей; возможность добавления новостей и их комментирования; загрузка медицинской и методической литературы; система оценивания и т.д.

Данный проект, как и любое веб-приложение, состоит из нескольких независимых частей: серверная часть, клиентская часть и система управления базы данных (СУБД).

Для реализации серверной части использовался язык программирования Ruby 1.8.7 и веб-фреймворк Ruby on Rails 3.2 [1]. Основной причиной выбора данной технологии явилось то, что фреймворк Rails позволяет максимально быстро разрабатывать веб-приложения за счет богатого набора различных средств разработки и дополнительных библиотек. Ruby on Rails включает в себя ряд библиотек, используемых для упрощения веб-разработки. В создании данного веб-проекта использовались компоненты: Active Record, Action View и Action Controller. Вместе эти библиотеки образуют MVC-архитектуру. Фреймворк Rails имеет чрезвычайно удобную оболочку для выполнения рутинных

действий, таких как создание контроллеров, моделей и выполнение миграций, модифицирующих схему базы данных. Помимо этого, разработчику предоставлен настроенный веб-сервер для тестирования проекта.

Клиентская часть использует стандартный набор языков: HTML, CSS и Java Script.

В качестве СУБД была использована PostgreSQL [2]. Основной причиной подобного выбора была её совместимость с облачной платформой для дальнейшего запуска приложения. Никаких специфических для данной СУБД возможностей не использовалось, и поэтому приложение может быть легко запущено и для работы с другими реляционными СУБД, такими как, например, MySQL, SQLite, Oracle или MS SQL. Процесс миграции с одной СУБД на другую достаточно прост благодаря механизму миграций, и основной объем работы в таком случае может занять только перенос самих данных.

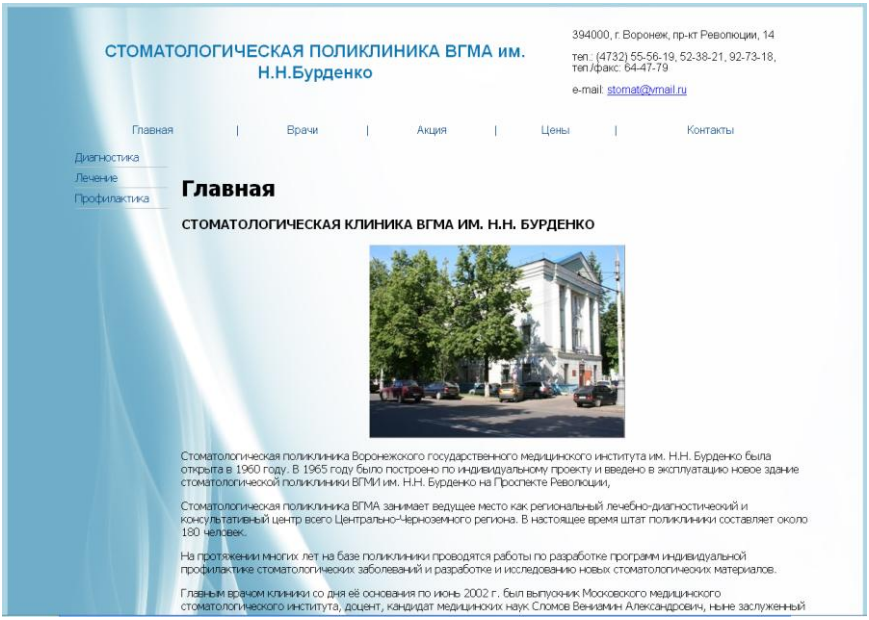

Рис.1. Интерфейс интернет-сайта поликлиники

Проект изначально разрабатывался с расчетом хостинга на платформе Heroku, представляющей собой облачный PaaS-сервис. Использование Heroku позволило полностью сконцентрироваться на разработке приложения, не задумываясь о таких инфраструктурных моментах, как настройка сервера приложений или проксирующего сервера, разработка схемы развертывания новых версий, продумывание схемы резервного копирования данных и возможного дальнейшего масштабирования приложения. Все перечисленные возможности предоставляются разработчикам облачным хостингом Heroku.

Повышение эффективности работы сайта может быть достигнуто следующими способами: регулярным обновлением информации и добавлением новых материалов; введением на сайт интерактивных элементов, которые могли бы упростить связь между администраторами сайта и его пользователями; проведением информационной кампании для привлечения внимания всех сотрудников и пациентов поликлиники к данному веб-сайту.

Рис.2. Одна из страниц интернет-сайта

Список литературы

1. Тейт Б. Ruby on Rails. Быстрая веб-разработка / Б. Тейт, К. Хиббс // СПб.: BHV-Петербург, 2008. – 224 с.
2. Уорсли Дж. PostgreSQL. Для профессионалов / Дж. Уорсли, Дж. Дрейк // СПб.: Питер, 2003. – 496 с.

РАЗРАБОТКА ИНФРАСТРУКТУРЫ ИНТЕРНАЦИОНАЛИЗАЦИИ JAVA-ПРИЛОЖЕНИЙ НА ОСНОВЕ ПРОЕКТА GNU GETTEXT

студ. А. А. Сбродов,
ст. преп. Д. И. Соломатин

Локализация пользовательского интерфейса – действительно важная часть процесса разработки, которая сильно влияет на популярность продукта. Это достаточно трудоемкая задача, требующая работы квалифицированных специалистов. При этом существует комплекс мер, автоматизирующий и упрощающий процесс локализации программного обеспечения (ПО), объединяемый понятием интернационализации [1, 2].
К сожалению, что касается стека технологий JAVA, то несмотря на высокую стоимость работ по локализации приложения и, зачастую, их неотвратимость, существующие инструменты интернационализации не развиты и обладают рядом существенных недостатков. Решение, созданное в рамках данной работы, не имеет описанной выше проблемы. Рассмотренный набор инструментов придерживается ряда базовых принципов интернационализации [3], которые в большинстве случаев игнорируются существующими решениями для платформы JAVA, как то:

1) ресурсы локализации отделены от бизнес-логики и пользовательского интерфейса;
2) приложение сохраняет функциональность при отсутствии ресурсов локализации;
3) код остается читаемым после интеграции с фреймворком;
4) ресурсы локализации содержат информацию о контексте использования, примеры, дающие возможность переводчику создать корректный перевод;
5) инструмент локализации делает возможной работу переводчика, обладающего базовой компьютерной грамотностью, без специальной подготовки;
6) для параметризованных ресурсов поддерживаются множественные формы для разных локалей;

7) один и тот же текст переводится только один раз.

Сегодня для поддержки интернационализации повсеместно используется ResourceBundle, и поэтом именно с ResourceBundle проводится сравнение разрабатываемой инфраструктуры. Несмотря на огромную популярность, для этого решения выполняются только первый и частично второй пункты из списка.

В рамках реализации в качестве базового формата хранения ресурсов используется формат gettext, используемый при написании многоязычных программ для Unix-подобных операционных систем инфраструктуры GNU gettext. Такое решение обосновано тем, что сама по себе эта система реализует все базовые подходы к интернационализации, имеет массу разнообразных инструментов для различных платформ. Кроме всего прочего GNU gettext за множество лет разработки определил отточенный эффективный рабочий процесс локализации. К сожалению, использование GNU gettext для JAVA-приложений возможно только через конвертирование ресурсов локализации в «бедный» формат ResourceBundle, что сводит на нет все его преимущества [4].

Рабочий процесс локализации идентичен процессу для GNU gettext и работает по схеме, представленной на рис. 1 [5].

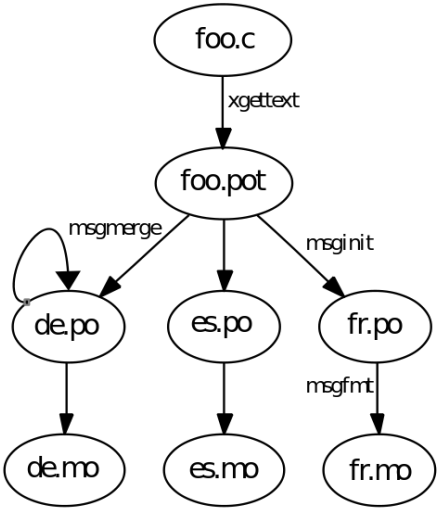

Рис.1. Схема процесса перевода с помощью gettext

Используются следующие форматы хранения ресурсов локализации:

1. .mo (англ. Machine Object) – бинарный файл, удобный для чтения программой и специфичный для платформы. Файлы этого типа добавляются к проекту и уже напрямую используются программой для получения локализованных сообщений в правильном виде;
2. .po (англ. Portable Object) – человеко-читаемый файл перевода, не зависящий от платформы. Файлы этого типа используются для редактирования переводов. Существует множество утилит позволяющих редактировать их содержимое, хотя то же самое можно делать вручную;
3. .pot (англ. PO template) – каталог, заготовка файла .po для перевода на новый язык. Этот файл может быть составлен вручную, но в большинстве случаев удобно использовать средства автоматической генерации [6].

Кроме самих строк перевода .po файлы могут содержать мета-информацию, которая не попадает в комментарии переводчика и различные служебные пометки.

Разработанное решение может работать со всеми этими типами файлов. Так, в реализации классы MoI18nResource и PoI18nResource предназначены для работы с файлами .mo и .po/.pot соответственно.

Каждая реализация источника ресурсов локализации имеет свое предназначение. Предполагается, что бинарный формат будет использоваться при нормальной работе системы в силу высокой производительности этого варианта, тогда использование .po-файлов для получения перевода отменяет необходимость при каждом изменении компилировать ресурсы локализации, что существенно увеличивает скорость отладки и тестирования вносимых изменений при разработке.

Помимо всего прочего, при разработке системы важным критерием высокой качественной оценки решения является как удобство использования с самого старта проекта, так и возможность безболезненного внедрения системы в уже существующие проекты. Таким образом, обеспечение обратной совместимости – одна из наиболее приоритетных задач работы.

Для решения этой проблемы была добавлена дополнительная реализация источника ресурсов локализации PropertiesI18nResources, которая служит оберткой над обычным классом ResourceBundle. Это позволяет использовать новый механизм загрузки переводов, не меняя сам формат хранения переводов. Конечно, в этом случае невозможно использовать несколько множественных форм, но для плавного перехода на полноценное использование разрабатываемого решения такой вариант является наиболее оптимальным.

Еще одним важным преимуществом реализованной инфраструктуры является ее универсальность за счет использования независимых модулей. Это сделано в силу того, что при проектировании системы важную роль играет факт, что далеко не всегда разработчик, использующий данное решение, нуждается во всех заявленных функциях системы на текущем этапе разработки.

Система содержит следующий перечень модулей:

1) **gettext4j-core** – модуль, содержащий набор базовых функций, включающие в себя инструменты загрузки ресурсов и получения локализованного текста из последних;
2) **gettext4j-toolkit** – модуль, включающий в себя логику управления переводами и автоматизации работы переводчика;
3) **gettext4j-toolkit-web** – модуль, представляющий из себя веб-приложение, являющееся пользовательским интерфейсом и оберткой над функциональностью gettext4j-toolkit;
4) **gettext4j-deployer** – модуль, содержащий логику для автоматических загрузки и обновления ресурсов локализации без остановки системы; также имеет зависимость от gettext4j-core и расширяет некоторые функции последнего;
5) **gettext4j-core** – модуль, включающий в себя набор библиотек для удобной работы с использованием Spring Framework.

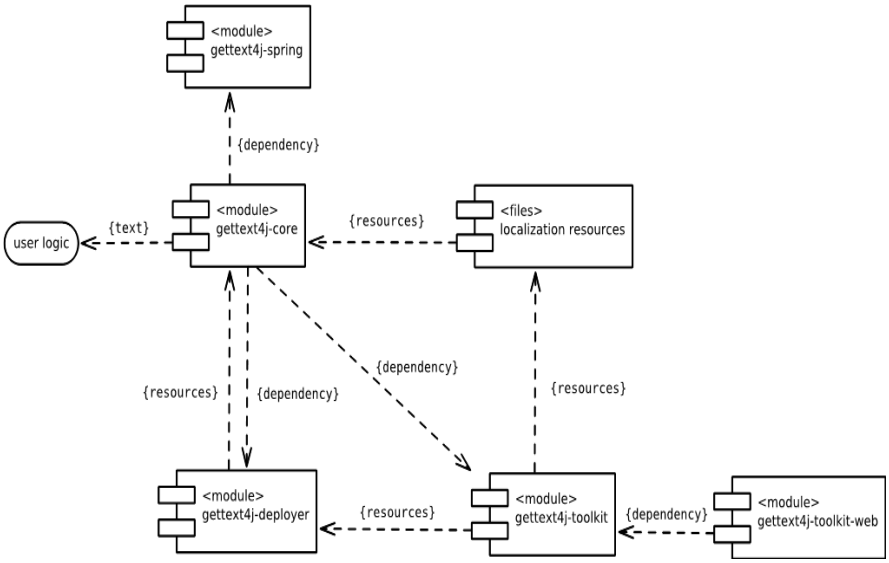

Рис.2. Структура системы

Благодаря такой структуре разработанное решение может использоваться как источник ресурсов локализации, в том числе, не создавая самих ресурсов, и как заглушку и задел на будущее. Также есть возможность подключить требуемые модули и использовать предоставляемую ими функциональность.

В итоге, исходя из всего выше сказанного, разработанную систему можно рекомендовать в качестве расширенной альтернативы малофункциональному набору инструментов ResourceBundle.

Список литературы

3. Mackenzie Adrian Internationalization: software, universality and otherness. Internationalization in Java / Adrian Mackenzie – 2006.
4. Gross Steffen Internationalization and Localization of Software / Steffen Gross – Approved at Ypsilanti, Michigan, 2006.
5. After Internationalization (The Java™ Tutorials). – Sun Microsystems
6. Wendy. K. Adams. Making On-Line Science Course Materials Easily Translatable and Accessible Worldwide: Challenges and Solutions / Wendy. K. Adams, Hisham Alhadlaq, Christopher V. Malley и др. // Journal of Science Education and Technology, Feb 2012.
7. Официальная документация GNU gettext. http://www.gnu.org/software/gettext/
8. Michael Kerrisk. The Linux Programming Interface: A Linux and UNIX System Programming Handbook / Michael Kerrisk – No Starch Press, 2011

ИСПОЛЬЗОВАНИЕ ТЕХНОЛОГИЙ НЕЙРО-КОМПЬЮТЕРНЫХ ИНТЕРФЕЙСОВ В ЛАБОРАТОРИИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ В МЕДИЦИНЕ ФКН

студ. А.Г.Семенов,
проф. С.Д. Кургалин,
доц. Я.А.Туровский

Нейро-компьютерные интерфейсы (НКИ), представляющие собой новый способ коммуникации человек-компьютер, используют различные методы регистрации и интерпретации феноменов, получаемых на

электроэнцефалограммах, магнитно-резонансных томограммах (МРТ) головного мозга, транскраниальных оксиметрах и т.д.

Специфика НКИ заключается в том, что при его работе команды или сообщения, посылаемые индивидуумом во внешний мир, идут от мозга не через обычные выходные каналы – периферийные нервы и мышцы, а регистрируются в виде электромагнитных сигналов, формирующихся в результате активизации отдельных нейронов или нейронных ансамблей. Существенное увеличение скорости работы систем НКИ при сохранении или, возможно, незначительном снижении точности является весьма актуальной задачей, решение которой имеет не только теоретическое, но и большое практическое значение и может обеспечить создание в будущем новых типов НКИ.

В настоящее время НКИ характеризуются относительно невысокой скоростью ввода данных и не обеспечивают пока их необходимой точности, если их сравнивать со стандартными компьютерными устройствами: клавиатурой, джойстиком, мышью и т.п. Это, прежде всего, связано с двумя проблемами, хорошо известными разработчикам систем биологической обратной связи (БОС). К первой из них относится, практически, полное исчерпание списка известных электрофизиологических и МРТ-феноменов головного мозга, которые можно было бы использовать для формирования новых команд управления. Вторая проблема связана с крайне высокой ресурсоёмкостью перспективных технологий обработки сигналов, на основе которых командная информация передается органам-эффекторам. Это делает их малопригодными для управления внешними устройствами в реальном масштабе времени. Помимо этих двух наиболее известных проблем, необходимо указать на ещё одну, мало исследованную, заключающуюся в том, что подавляющее большинство современных систем НКИ не учитывает роль человека в иерархической системе принятия управленческих решений. Современные технологии НКИ подразумевают всё более активную роль человека на всех этапах процесса управления, что, однако, существенно замедляет скорость человеко-машинного взаимодействия. Для ускорения работы НКИ следует оставить пользователю только выбор общих стратегий поведения, а программно-аппаратное обеспечение НКИ должно реализовывать тот алгоритм действий, который будет наиболее эффективным, оптимальным, и, что самое главное, отражать особенности управления конкретного человека [1].

Кроме того, оценивая результаты разработки систем НКИ, можно сделать вывод, что, несмотря на большой эмпирический материал,

накопленный как в этой области исследований, так и в смежных областях, в частности, при создании систем биологической обратной связи, в рамках НКИ пока ещё существует немного моделей человеко–машинного взаимодействия, имеющих прогностическую ценность. Прогнозирование эффективности работы НКИ и учёт индивидуальных особенностей пользователя позволит существенно приблизить технологию НКИ к конечному потребителю. В этой связи существенную роль в создании подобных моделей НКИ может сыграть использование активностной психофизиологической парадигмы [2], постулирующей наличие у управляющего объекта модели результата действия ещё до начала самого действия. Для создания наиболее адаптированного для пользователя нейро-компьютерного интерфейса по асинхронной неинвазивной схеме, в первую очередь, можно продублировать иерархическую схему управления, реализованную в центральной нервной системе высших животных. В этом случае произойдёт разделение функций, которые в НКИ выполняет (в ограниченном виде) только компьютер. Пока ещё в системах НКИ, практически, не используются технология «опережающего отражения», учитывающего прогноз изменения окружающей обстановки, отсутствует также и «акцептор результата действия», т.е. модель полезного результата. Иными словами, компьютерная часть современных НКИ работает при отсутствии информации о требуемом результате действия, и, во многом, о динамике изменения окружающей среды. Поэтому перспективным является развитие технологий НКИ на основе анализа фоновой активности головного мозга с использованием модели «опережающего отражения».

В лаборатории информационных технологий в медицине кафедры цифровых технологий факультета компьютерных наук Воронежского государственного университета разрабатываются и внедряются технологии НКИ, направленные на решение указанных выше проблем, и на их основе создаются новые методики для совершенствования учебного процесса и развития научных исследований.

При использовании нейро-компьютерных интерфейсов учитывается: а) связь базовой парадигмы НКИ только лишь с оценкой электроэнцефалограмм (ЭЭГ) для асинхронных систем, т.е. систем, где темп и характер передачи информации задаёт человек, с последующей трансформацией данных ЭЭГ в команды для устройств-эффекторов, при этом индивидуальные паттерны поведения человека и его текущее положение в многомерном пространстве поведенческих реакций, практически, не учитываются при оценке того, что именно пользователь хочет получить в данный момент от устройств–эффекторов; б) несовершенство современных методов обработки фоновых ЭЭГ, что не

позволяет достоверно выделить устойчивые паттерны мозговой активности человека; в) малую доступность инвазивных (т.е. связанных с имплантацией электродов) интерфейсов и недостаточную изученность механизмов взаимодействия электродов с нервной таканью; г) заметную зависимость неинвазивных интерфейсов от места расположения электродов; д) существенные ограничения, накладываемые на выделение вызванных потенциалов головного мозга из общей структуры сигнала ЭЭГ [3] для синхронных систем, когда темп и характер передачи информации по НКИ задаёт компьютер.

В лаборатории создана линейка человеко-машинных интерфейсов [4], в их составе система управления внешними устройствами движением глаз или головы; биоуправляемый протез – анализ электромиограмм позволяет преобразовать электрическую активность мышц в команды компьютера или движения механизмов протеза. Проводится подготовка к подключению биопротеза непосредственно к нервам конечности.

Создается единый программно-аппаратный комплекс, включающий в себя, наряду с разработками в области нейро-компьютерных и человеко-машинных интерфейсов, и новые программные решения.

Разрабатываемые технологии НКИ используются при подготовке IT-специалистов, осуществляемой на факультете компьютерных наук Воронежского госуниверситета.

Список литературы

1. Кургалин С.Д. Оптимизация работы нейро-компьютерного интерфейса с учетом поведения человека / С.Д. Кургалин, Я.А. Туровский, А.В. Максимов // Вестник Воронеж. гос. ун-та. Сер. Системный анализ и информационные технологии. – 2012. – № 1. – С. 135-139.

2. Туровский Я.А. Моделирование нейрокомпьютерного интерфейса на основе активностной парадигмы / Я.А. Туровский, С.Д.Кургалин, А.В. Максимов // Системы управления и информационные технологии. – 2012. – № 1(47). – С. 99-103.

3. Туровский Я.А. Интерфейс мозг-компьютер, использующий амплитудный анализ зрительных вызванных потенциалов / Я.А. Туровский, С.Д.Кургалин, А.Г. Семенов // Информатика : проблемы, методология, технологии : материалы XIII Международ. науч.-метод. конф. – Воронеж, 2013. – Т. 3. – С.298-302.

4. Кургалин С.Д. Линейка человеко-машинных интерфейсов / С.Д. Кургалин, Я.А. Туровский, А.В. Максимов // Всероссийский конкурс научно-исследовательских работ студентов и аспирантов в области информатики и информационных технологий : сб. науч. работ : в 3 т. – Белгород, 2012. – Т. 2. – С. 493-495.

РАЗРАБОТКА СИСТЕМЫ КОЛЛЕКТИВНОГО ПРИНЯТИЯ РЕШЕНИЙ

студ. А.А. Сидоркин,
доц. Тюкачев Н.А.

Задача принятия решений (ЗПР) — одна из самых распространенных в любой предметной области. Ее решение сводится к выбору одной или нескольких лучших альтернатив из некоторого набора. Для того чтобы сделать такой выбор, необходимо четко определить цель и критерии (показатели качества), по которым будет проводиться оценка некоторого набора альтернативных вариантов. Выбор метода решения такой задачи зависит от количества и качества доступной информации. Данные, необходимые для осуществления обоснованного выбора, можно разделить на четыре категории: информация об альтернативных вариантах, информация о критериях выбора, информация о предпочтениях, информация об окружении задач [1].

Одним из наиболее универсальных, удобных и математически обоснованных методов, является метод анализа иерархий (МАИ). Он предполагает декомпозицию проблемы на все более простые составляющие части и обработку суждений лица, принимающего решение. В результате определяется относительная значимость исследуемых альтернатив для всех критериев, находящихся в иерархии. Относительная значимость выражается численно в виде векторов приоритетов. Полученные таким образом значения векторов являются оценками в шкале отношений и соответствуют так называемым жестким оценкам [1].

У метода анализа иерархий, существует множество различных модификация, но общая схема состоит в следующем:

9. Постановка задачи выбора;
10. Выбор критериев;
11. Выбор альтернатив;

12. Построение матрицы попарных сравнений;
13. Вычисление векторов приоритетов для критериев и оценка согласованности матриц;
14. Вычисление глобальных векторов приоритетов;
15. Из глобальных векторов приоритетов следует наиболее приемлемая (согласно данному методу) альтернатива.

Например, необходимо оценить богатейшую ресурсами страну (рис. 1), для этого необходимо составить список ресурсов (золото, нефть, газ и т.д.). Далее эксперту необходимо оценить значимость каждого ресурса для мировой экономики относительно других, построив матрицу парных сравнений.

Из этих матриц высчитывается вектор приоритетов, который покажет общую значимость каждого ресурса. Следующим шагом является оценка относительного богатства стран для каждого ресурса, например, оценка богатства стран газом относительно друг друга. Для этого также необходимо построить матрицы попарных сравнений и вычислить вектор приоритетов. Далее на основе вектора приоритетов значимости каждого ресурса, а также на основе векторов богатства стран каждым ресурсом, необходимо вычислить глобальный вектор приоритетов, веса которого определят место каждой страны в мировой экономике.

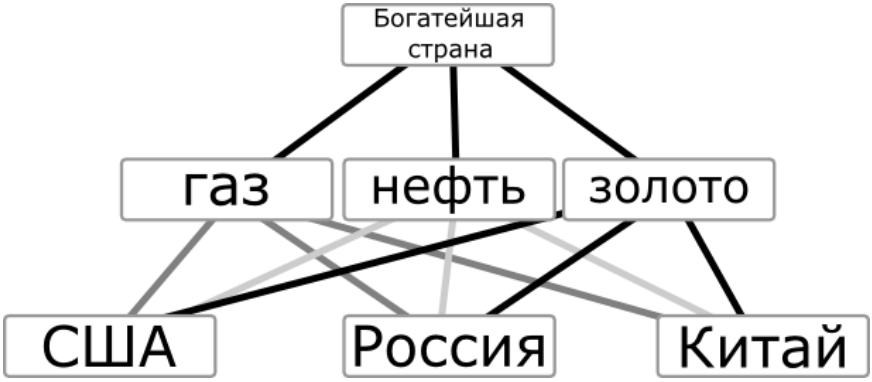

Рис 1. Иерархия для выбора богатейшей страны

Рассмотренный случай относительно простой, так как иерархия трехуровневая и однородная, но это не всегда так. Данный пример хорошо демонстрирует суть метода анализа иерархий.

Как правило, решение принимает не один эксперт, а несколько. Результаты работы экспертов можно обобщить на основе средне геометрического.

Для принятия решения методом анализа иерархий необходимо проводить много, хотя и простых, расчетов. Неудивительно, что для этого написано много компьютерных программ, однако приложений позволяющих комфортно и независимо работать нескольким экспертам, найдено не было. Также не существует приложений, позволяющих интегрировать различные методы принятия решений.

В данной статье рассматривается следующая задача: создание системы совместного принятия решения несколькими экспертами на основе метода анализа иерархий с возможностью расширения другими методами.

Рассмотрим другие задачи и требования к системе.

Система в развернутом виде (например, на предприятии) предназначена для ограниченного круга лиц, поэтому должна быть обеспечена безопасность, а также контроль администратором пользователей. Если необходимо предоставить систему для открытого использования, важно учесть возможность регистрации пользователей.

Важным требованием к системе также является учет и разграничение ролей пользователей, таких как администратор, менеджер проектов, рядовой пользователь. Предполагается, что менеджер проектов может создавать проект, назначать пользователей, утвердить иерархию (или альтернативы, а построение иерархии оставить пользователям), учесть политику принятия решений (анонимные или открытые оценки).

Также важно определить какие модификации метода анализа иерархий, и какие другие методы изначально будет поддерживать система, но принятие данного решения будет выполнено на более поздней стадии проекта.

Рассмотрим предполагаемую схему работы данного приложения с методом анализа иерархий:

1. Администратор регистрирует пользователей и высылает им данные по электронной почте (автоматически через интерфейс приложения);

2. Пользователь с ролью менеджера проектов создает проект, строит иерархию (или определяет альтернативы), назначает экспертов на проект;
3. Эксперты выставляют свои оценки и отправляют результат;
4. Менеджер получает уведомление и смотрит результат работы экспертов, вычисленный с помощью средне геометрического.

Следующим этапом необходимо определиться с тем, какими технологиями будет достигнут результат. Во-первых, данное приложение должно быть выполнено в виде веб-приложения, преимущества такого подхода вполне очевидны, мобильность, простота обновления и др. Также у данного приложения, должен быть богатый и удобный пользовательский интерфейс, т.е. соответствовать уровню RIA (Rich Internet Application). Это может быть достигнуто несколькими способами, например, можно использовать одну из множеств JavaScript библиотек (qooxdoo, ExtJS и пр.) и организовать обмен данными с сервером в виде HTTP RPC-API, на основе REST, SOAP и др. архитектур. Также вместо JavaScript можно использовать Flex. Преимущество данного подхода в том, что не имеет значения на каком языке разрабатывать серверное приложение, однако, разработки такого уровня приложения, средствами javascript (и в меньшей степени ActionScript) затруднительны. Другой вариант - это использовать единую технологию (язык) для создания данной системы. Наиболее богата данными технологиями платформа Java, к данной платформы существует множество фреймворков, таких как: Eclipse RCP, echo, GWT, ZK, Vaadin и др. При оценки данных технологий наилучшим образом показал себя Vaadin, сочетающий в себе удобный, грамотно построенный API и богатые возможности. Данный фреймворк позволяет разрабатывать enterprise веб-приложения, почти также легко, как и приложения для рабочего стола.

Краеугольным камнем любого приложения является СУБД и фреймворк для работы с ней. На сегодняшний день разнообразие технологий настолько велико, что можно легко запутаться. Помимо наиболее популярных реляционных СУБД существуют документо-ориентированные, графовые, семантические, типа ключ-значение, объектно-ориентированные и пр. Также почти на каждый вид СУБД, существуют десятки различных реализаций. Для данного приложения, было принято решения использовать объектно-ориентированную СУБД db4o. Данная СУБД позволяет организовать структуру в сущностях языка Java, таким образом, избавляя от необходимости использовать другие концепции и языки отличные от Java.

Важным аспектом системы является ее грамотно построенная архитектура. Разрабатывая приложения, важно делать их удобными для

поддержки, то есть писать код грамотно и понятно. Хотя точного ответа как нужно писать код не существует, в данной статье будут рассмотрены некоторые принципы разработки ПО. Ключевой момент в данного рода приложениях – неоправданная сложность архитектуры. За многие годы практики специалистами были выработаны шаблоны проектирования, хотя зачастую они используются неправильно и только вносят ненужную сложность. Например, шаблон «Абстрактная фабрика», имея идею простой замены реализации, вносит неопределенность в код и затрудняет его чтение таким образом, что порой очень сложно добраться до логики самого приложения. Наиболее полезным шаблоном с минимумом риска является шаблон «MVC», который позволяет хорошо отделить различные аспекты приложения друг от друга. В контексте данного приложения модели – классы базы данных, отображения – компоненты Vaadin, а контроллеры – команды. Также полезным шаблоном является Dependency Injection, который позволяет избавиться от проблемы зависимости объетов друг от друга, а еще является отличной альтернативой абстрактным фабрикам. В качестве реализации в данном приложении используется библиотека Google Guice как наиболее удобная реализация данного шаблона и, в отличие от Spring, Google Guice нуждается в минимальной конфигурации.

В качестве подведения итога, можно сказать, что в данной статье была рассмотрена постановка задачи создания системы принятия решений, а также были проведены рассуждения о пути ее решения.

Список литературы

1. Андрейчиков А.В. Анализ, синтез, планирование решений в экономике / А.В. Андрейчиков, О.Н. Андрейчикова // М.: Финансы и статистика, 2002. - 368с.

2. Саати Т. Принятие решений / Т. Саати // Метод анализа иерархии: Пер. с англ. – М.: Радио и связь, 1993. – 320 с.

НЕЙРОСЕТЕВЫЕ ТЕХНОЛОГИИ СОЗДАНИЯ ЦИФРОВЫХ ВОДЯНЫХ ЗНАКОВ В АУДИОФАЙЛАХ

студ. Н. Н. Ускова,
асс. Е. Ю. Митрофанова

Мультимедийные файлы часто подвержены копированию и нелегальному распространению, что влечет за собой нарушение авторских прав. Чтобы защитить авторские права стали применяться технологии цифровых водяных знаков (ЦВЗ). ЦВЗ представляют собой специальные метки, встраиваемые в объекты с целью их аутентификации и защиты от нелегального использования. Для внедрения и скрытия невидимых ЦВЗ используются различные стеганографические методы скрытия информации [1].

Одной из технологий стеганографического скрытия данных является использование аппарата искусственных нейронных сетей (ИНС).

Целью работы является исследование нейросетевых алгоритмов создания и извлечения цифровых водяных знаков в аудио-файлах при изменении различных характеристик сигнала, внесении шума, проведении фильтрации сигнала и анализ наиболее эффективных вариантов внедрения ЦВЗ.

Постановка задачи стеганографического скрытия информации с помощью ИНС может быть сформулирована следующим образом: требуется с использованием функциональных возможностей нейронных сетей для любого фрагмента контейнера, представленного в виде вектора z, и вектора встраиваемых данных d, построить отображения [2]

$$\begin{aligned} &\tilde{z} = F_1(z), \tilde{z} \in \tilde{Z},\ \bar{z} = F_2(\tilde{z}, d),\ \bar{z} \in \bar{Z},\ \|\bar{z} - z\| \to \min, \\ &\tilde{d} = F_3(\bar{z}), \tilde{d} \in D, \left\|d - \tilde{d}\right\| \to \min, \end{aligned} \tag{1}$$

где первый оператор *F1* реализует сжимающее отображение входных данных, обеспечивающее подготовку вектора-контейнера к встраиванию данных, оператор *F2* реализует собственно встраивание ЦВЗ, а оператор *F3* – восстановление ранее скрытой информации.

При этом должны выполняться следующие требования к системам внедрения ЦВЗ:

- скрываемая информация не должна вносить в сигнал искажений, воспринимаемых системой слуха человека;
- скрываемая информация должна быть стойкой к наличию шумов, фильтрации;
- скрываемая информация не должна вносить заметных изменений в статистику контейнера.

Общая структура алгоритма представлена на рис. 1.

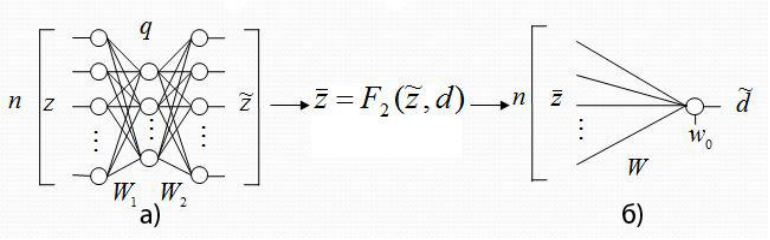

Рис. 1. Общая структура работы алгоритма а) нейронная сеть, реализующая сжимающее преобразование б) нейронная сеть, реализующая восстановление скрытой информации

Для подготовки контейнера к встраиванию используется двухслойная нейронная сеть, представленная на рис. 1.а. При этом количество нейронов в скрытом слое нейронной сети $q<n$. Для реализации алгоритма встраивания данных сначала проводится обучение сети, обеспечивающее сжимающее отображение. Обучение проводится по совокупности реализаций входного вектора z и целевого векторов $\tilde{z}$ так, чтобы минимизировать величину средней квадратичной ошибки представления входного вектора на выходе сети. В результате обучения нейронной сети получается индивидуальное для данного набора данных сжимающее отображение с весьма незначительными потерями, при котором, получаемый на выходе сети вектор $\tilde{z}$, может быть представлен в виде разложения по первым q собственным векторам.

С целью минимизации ошибки искажения контейнера будем использовать сеть, для которой $q=n-1$, тогда $\tilde{z}$ будет отличаться от z только «высокочастотной» составляющей с малой амплитудой и дисперсией соответствующей минимальному собственному числу выборочной матрицы ковариации.

После того, как первая сеть обучена, выполняется оператор F_2, реализующий встраивание ЦВЗ в файл контейнер, где в качестве ЦВЗ

будем рассматривать двоичную последовательность d, Для этого сначала рассчитывается минимальное собственное число r_{min} по высокочастотной составляющей сигнала:

$$r_{\min} = \frac{1}{P} \sum_{p=1}^{P} (z^{(p)} - \tilde{z}^{(p)}) \quad (2)$$

Затем используя собственное число r_{min}, вычисляется нормированный вектор

$$\varphi_n = r_{\min} \Big/ \sqrt{\left(r_{\min}^{\ T} r_{\min}\right)}, \quad (3)$$

который затем используется для встраивания скрытой последовательности $d^{(p)}$

$$\bar{z}^{(p)} = \tilde{z}^{(p)} + a_m d^{(p)} \varphi_n \ , \quad (4)$$

где a_m – амплитуда вносимого искажения.

Для восстановления таким образом скрытых данных, может быть использована однослойная линейная ИНС, показанная на рис. 1.б.

Экспериментальный анализ разработанного алгоритма проводился на аудиофайлах формата wav из-за полного представления сигнала в этом формате без сжатия [3].

Алгоритм реализован в программной среде Matlab.

Основные характеристики аудио, которые могут влиять на работу алгоритма – это частота дискретизации, битность амплитуды сигнала и количество каналов воспроизведения.

Для преобразования данных использовалась двухслойная линейная нейронная сеть с функцией обучения trainscg, обучение сети проводилось на 8000 эпох. Для восстановления данных была использована линейная однослойная нейронная сеть, обученная с помощью 2000 эпох обучения.

Результаты работы алгоритма при изменении частоты дискретизации сигнала показаны на рис.2.

На рис. 2 используются следующие обозначения: SQ-ob – среднеквадратичная ошибка искажения контейнера при тестировании на данных, используемых для обучения нейронной сети; SQ-ts – среднеквадратичная ошибка искажения контейнера при тестировании на данных, не участвовавших в обучении нейронной сети; Pex-ob – оценка вероятности ошибки при восстановлении ЦВЗ, при тестировании на данных, используемых для обучения нейронной сети; Pex-ts – оценка вероятности ошибки при восстановлении ЦВЗ при тестировании на данных, не участвовавших в обучении нейронной сети.

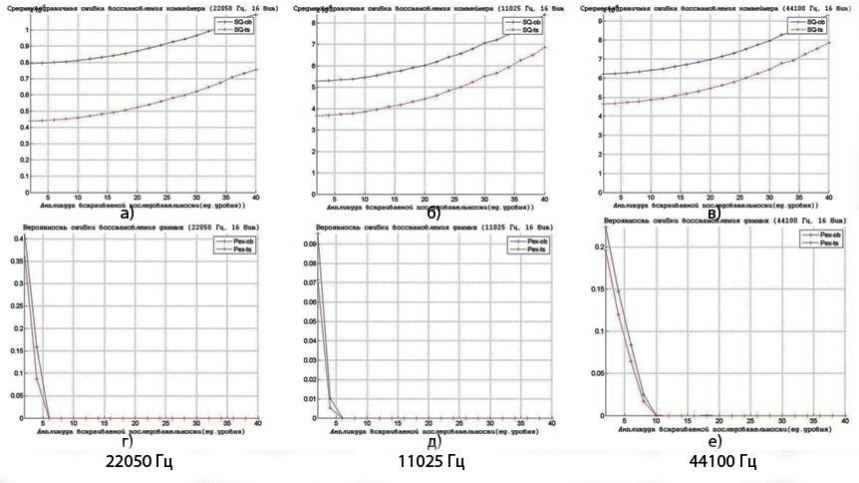

Рис. 2. Среднеквадратическая ошибка восстановления данных а) 22050 Гц; б) 11025 Гц; в) 44100 Гц; Вероятность ошибки восстановления данных г) 22050 Гц; д) 11025 Гц; е) 44100 Гц

Как видно из графиков, что среднеквадратичная ошибка и для тестовых данных, и для обучающей выборки имеет одинаковую форму в зависимости от амплитуды для всех частот. При этом сохраняется одинаковая зависимость увеличения среднеквадратической ошибки при увеличении амплитуды.

Вероятность ошибки восстановления данных с увеличением амплитуды резко уменьшается, и становится равной нулю через несколько отсчетов, это обеспечивает практически полное качественное восстановление встраиваемой информации и показывает, что нейронная сеть достаточно быстро обучается.

Алгоритм был протестирован на 8 различных частотах (кГц): 4, 8, 11, 22, 44,1, 32, 48, 96. Все величины проверены и для обучающей выборки и для тестовых данных. Для всех этих частот наблюдается одинаковая стабильность и идентичность работы алгоритма.

Из этого исследования можно сделать вывод, что алгоритм способен обрабатывать сигналы с различными частотами дискретизации, при этом результаты работы в зависимости от частоты не изменяются.

Аудиофайлы имеют различную битность сигнала, от которой зависит какое количество бит могут занимать значения амплитуды.

Для исследования алгоритма встраивания ЦВЗ в аудио-файлы при различных значениях битности сигнала будут использоваться аудио-файлы со следующими характеристиками: частота дискретизации - 22 050Гц, моно сигнал. Для следующих уровней битности: 8 бит, 16 бит, 24 бит, 32 бит.

На рис. 3 показаны результаты характеристик абсолютной ошибки восстановления контейнера и вероятности ошибки восстановления данных. На графиках используются следующие обозначения: Delt-ob – абсолютная ошибка искажения контейнера при тестировании на данных, используемых для обучения нейронной сети; Delt-ts – абсолютная ошибка искажения контейнера при тестировании на данных, не участвовавших в обучении нейронной сети.

Вероятность ошибки восстановления данных для 16 (рис. 3.в) и 8 (рис. 3.г) бит с увеличением амплитуды резко уменьшается, и через несколько отсчетов становится равной нулю - это обеспечивает практически полное качественное восстановление встраиваемой информации.

Для 8 бит значения абсолютной ошибки (рис. 3.б) для обучающей выборки и тестовых данных совпадают, это связано с разбросом данных амплитуды. Для 8 бит значения амплитуды могут принимать значения в диапазоне $\{-2^8..2^8\}$. Так как амплитуда занимает меньше бит, поэтому разброс значений находится примерно на одном уровне относительно количества отсчетов. Разброс значений для большей битности значительно больше, поэтому для этих характеристик результаты для обучающей выборки и тестовых данных по значениям не совпадают, но совпадают по форме зависимости от значений амплитуды.

Для битности 8 бит для обучения было взято малое количество эпох обучения и НС показала качественное восстановление с минимальной вероятностью ошибки. Для большей битности для более точного восстановления требуется большое количество эпох обучения, для точного восстановления для 16 бит – 2000 эпох, для 24 бит – 4000 эпох, для 32 – 8000 эпох. При этом вероятность ошибки восстановления данных с увеличением битности увеличивается.

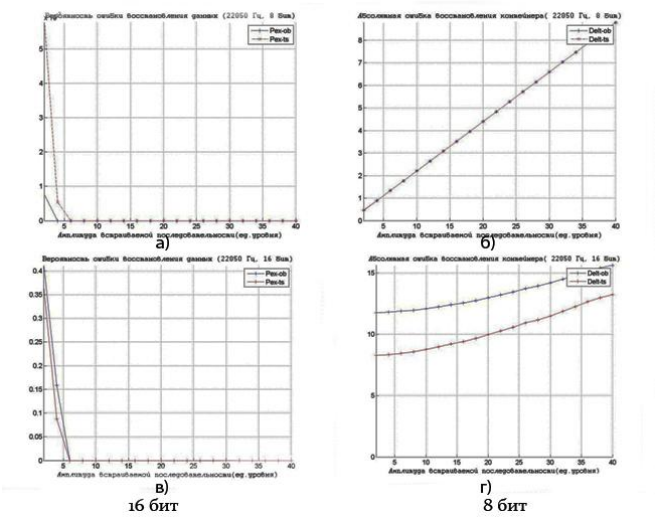

Рис. 3. Абсолютная ошибка восстановления данных а)16 бит; б) 8бит;

Вероятность ошибки восстановления данных в)16 бит; г) 8 бит

Для исследования на устойчивость к шуму и фильтрации были выбраны 2 цифровых фильтра – фильтр скользящего среднего и фильтр нижних частот(ФНЧ). Стояла задача создать оптимальные фильтры, чтобы они убирали нежелательные шумы, при этом минимально искажая исходный сигнал.

Путем экспериментов такие характеристики фильтров были подобраны.

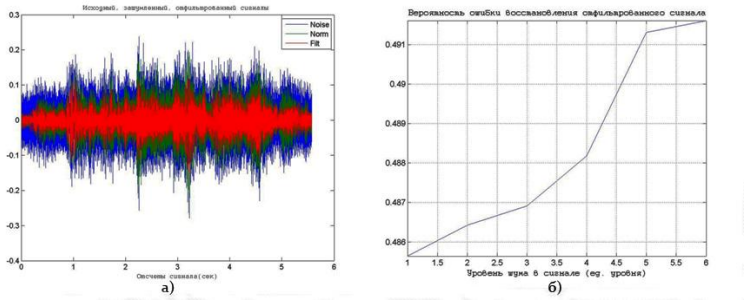

Рис. 4. Использование ФНЧ а) на зашумленном сигнале; б) Вероятность ошибки восстановления отфильтрованного сигнала

Как видно на рис. 4.а, шум был отфильтрован и в исходном сигнале полезный сигнал практически не пострадал. Было взято 10 различных уровней шума, как видно из рис. 4.б с увеличением шума вероятности ошибки восстановления так же увеличиваются, при этом, если брать большее количество эпох обучения для восстанавливающей НС, то эффективность восстановления увеличивается.

Для фильтра скользящего среднего так же путем экспериментов было подобрано оптимальное скользящее окно. Результаты работы алгоритма после фильтра скользящего среднего аналогичны результатам для ФНЧ.

Заключение

Были проведены работы по разработке и исследованиям нейросетевого алгоритма встраивания ЦВЗ в аудио файлы. Были проведены многосторонние исследования на изменения различных характеристик аудиофайлов, для исследования было взято большое количество тестовых ресурсов.

Разработанный алгоритм использования нейросетевых технологий встраивания ЦВЗ отвечает предъявляемым требованиям:

- способен обрабатывать сигналы с разными значениями частоты дискретизации, разрядности амплитуды сигнала, количеством каналов;

- показывает хорошие результаты восстановления с минимальными ошибками восстановления для всех случаев изменения характеристик сигнала, при этом не искажая исходных сигнал;
- показывает эффективную работу при работе с зашумленными сигналами и хорошее восстановление при обработки сигнала цифровыми фильтрами.

Список литературы

1. Грибунин, В. Г. Цифровая стеганография / В. Г. Грибунин, И. Н.Оков, И. В. Туринцев. – М.: Солон-пресс, 2002. - 272 с.

2. Сирота, А.А. Нейросетевые технологии создания цифровых водяных знаков / А.А. Сирота, М.А. Дрюченко, Е.Ю. Митрофанова // Нейрокомпьютеры. -2012. - №10. - С.13-20.

3. Rundow, A. Audio File Formats - http://composerfocus.com/audio-file-formats-a-rundown/

4. Использование оконных функций в задачах цифрового спектрального анализа - http://www.dsplib.ru/content/winex/winex.html

РАЗРАБОТКА СИСТЕМЫ УЧЕТА ВОЛС

студ. А. Н. Казанцев,
студ. А. И. Чекмарев,
ст. преп. Д. И. Соломатин

1. Задача учета ВОЛС

В настоящий момент высокоскоростные сети передачи данных, как правило, строятся на основе волоконно-оптических линий связи (ВОЛС). При построении крупных сетей связи в определенный момент обязательно возникает задача учета и паспортизации сети, в том числе ее пассивной составляющей – ВОЛС, а именно участков кабеля, оптических муфт, кроссов, коммутации волокон в муфтах и кроссах, а также физического расположения вышеуказанных объектов.

В существующих решениях (Кросс ПРО, UserSide и др.) часто не хватает тех или иных возможностей, кроме того, интеграция таких решений в существующую информационную систему либо невозможна (требуется перестраивать всю работу организации на новом программном

обеспечении) либо неоправданно трудоемка. Учитывая вышеизложенные соображения, более удобным решением является разработка собственной системы паспортизации ВОЛС, учитывающей специфику работы конкретной организации.

Помимо базовых возможностей систем данного типа был выделен ряд специфических требований:

- возможность учета как фактического состояния объектов сети, так и планируемых изменений;
- учет исполнителей всех работ по построению сети;
- интеграция с существующей информационной системой (здания, активное оборудование, клиенты и т.п.);
- разработка системы в виде web-приложения (работа с системой должна осуществляться в web-браузере).

2. Инструменты разработки

В качестве серверного языка разработки был выбран PHP версии 5.3, клиентской части – JavaScript, в качестве СУБД – Oracle версии 11g.

Передача данных между клиентом (web-браузер) и сервером осуществляется посредством AJAX-запросов в формате JSON.

При построении клиентской части использованы следующие JavaScript-библиотеки:

- jQuery – для работы с объектной моделью HTML-документа и асинхронного обмена данными с сервером;
- jQuery EasyUI (библиотека элементов управления) – для построения пользовательского интерфейса;
- Leaflet.js [1] – для работы с картой (различные источники карт, поддержка слоев, удобный API и т.д.);
- Kinetic.js [2] – для объектно-ориентированной работы с графикой HTML5 (рисование объектов, их перемещение, обработка событий уровня объектов и т.п.).

В качестве карт выбраны карты OpenStreetMap (открытость формата, возможность поднять свой собственный OpenStreetMap-сервер).

3. Картографический модуль

В разрабатываемой системе можно выделить два основных модуля – картографический модуль и модуль коммутации волокон.

Картографический модуль работает со следующими объектами предметной области:

- здания;
- точки на карте:
  - опоры (столбы);
  - колодцы кабельной канализации;
  - точки входа в здание;
- участки кабеля (проходит через точки на карте).

Картографический модуль обладает следующими основными возможностями:

- отображение карт из различных источников данных;
- отображение поверх карт контуров зданий, отсутствующих на карте, но заданных в базе данных;
- создание и редактирование на карте точек различных типов с описанием их параметров;
- создание и редактирование участков кабеля (с возможностью разрыва участка кабеля в точке);
- поиск зданий на карте по адресу;
- поиск объектов провайдера (кабели, опоры, колодцы и пр.) по названию и ключевым параметрам;
- привязка объектов (прежде всего точек входа в здание) к зданиям (это необходимо, т.к. к зданию может быть привязана различная информация из других систем (список клиентов, список оборудования, список заявок и т.п.);
- поддержка разделение всех объектов на планируемые и фактические.

В реализации данного модуля активно использовалась поддержка слоев в библиотеке Leaflet.js. Так выделены следующие слои:

- слой фактических объектов;
- слой планируемых объектов;
- слой объектов, планируемых для удаления.

Для осуществления привязки объектов к зданиям был проведен импорт контуров зданий из данных OpenStreetMap, а также их сопоставление по адресам с существующим у провайдера списком зданий. Для автоматической привязки точки к зданию был использован алгоритм проверки принадлежности точки многоугольнику [3].

На рис. 1 представлен интерфейс картографического модуля.

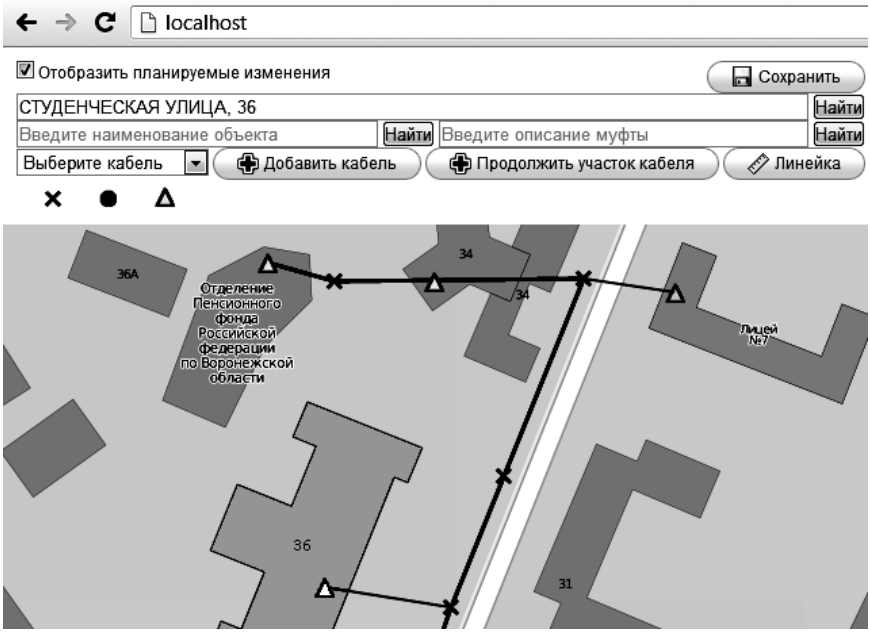

Рис. 1. Картографический модуль

4. Модуль коммутации волокон

Картографический модуль работает со следующими объектами предметной области:

- участки кабеля;
- оптические муфты;
- оптические кроссы;
- делители.

Делители бывают двух основных типов: делители мощности и делители частоты. Первые распределяют оптическую мощность в определённом процентном соотношении. Вторые могут разделять (или, наоборот, соединять) световой поток по частоте. Модуль коммутации волокон поддерживает оба типа делителей.

Основные задачи, решаемые модулем:

- графическое представление и редактирование коммутации волокон в соединительных устройствах (на схеме должны быть отображены кабели, делители, а также соединения волокон в виде линий);
- табличное представление коммутации волокон в оптических муфтах (с указанием фактического состояния и планируемых изменений; может использовать в качестве нарядов фактическим исполнителям работ по разварке оптики);
- хранение истории изменений коммутации волокон с фактическими исполнителями работ;
- возможность хранения как фактического состояния, так и планируемых изменений.
- учет контейнеров (шкафы и стойки), в которых могут располагаться оптические муфты и кроссы.

На рис. 2 представлен интерфейс модуля коммутации волокон.

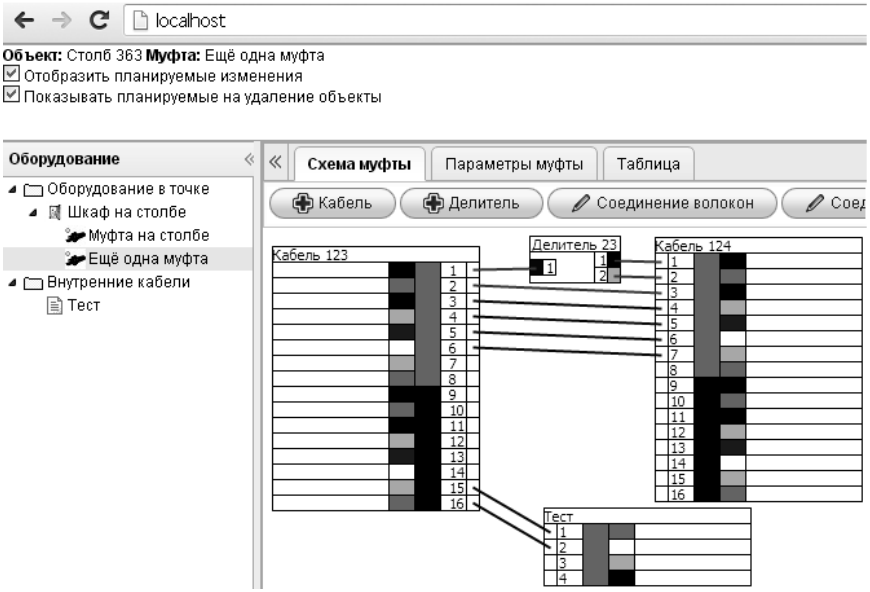

Рис. 2. Модуль коммутации волокон

5. Заключение

В настоящий момент в разрабатываемом проекте реализованы основные функции, необходимые для учета ВОЛС.

В перспективе планируется значительная доработка проекта в части автоматизации многих задач, которые возникают в процессе развития и эксплуатации волоконно-оптических сетей передачи данных.

Список литературы

1. Документация Leaflet.js. http://leafletjs.com/reference.html
2. Документация Kinetic.js. http://kineticjs.com
3. W. Randolph Franklin. PNPOLY – Point Inclusion in Polygon Test. http://www.ecse.rpi.edu/~wrf/Research/Short_Notes/pnpoly.html

АВТОМАТИЗАЦИЯ УПРАВЛЕНИЯ ПРОЕКТАМИ

студ. А.О. Шевляков,
проф. М. Г. Матвеев

Введение

Управление проектами зародилось в 30-60-х годах XX века и продолжает развиваться по сей день. Управление проектами – раздел теории управления социально-экономическими системами, изучающий методы, формы, средства наиболее эффективного и рационального управления изменениями [1].

На протяжении многих веков человечеству приходилось реализовывать множество проектов. Возрастающая сложность проектов, с одной стороны, и накопленный опыт управления, с другой, сделали необходимым и возможным создание идеологии и методологии управления проектами. Бурное развитие кибернетики, теории управления и исследования операций в середине 20 столетия позволило создать ряд формальных моделей и тем самым заложить систематическую научную основу управления проектами.

Одно из направлений изучения данной дисциплины - это модели и методы сетевого планирования, позволяющие определить рациональную или оптимальную последовательность выполнения работ при заданных

технологических, бюджетных и других ограничениях. Такого рода модели получили всеобщее признание, легли в основу многочисленных прикладных программ для ПК и широко используются для управления реальными проектами.

Сетевой анализ — это метод планирования работ проектного характера, т.е. работ, операции в которых, как правило, не повторяются [2]. Анализ такой сети обеспечивается той или иной сетевой технологией, которая представлена соответствующим комплексом математических моделей и методов.

Анализ проекта осуществляется в несколько этапов [2,3]:

1) проект разбивается на ряд операций (работ);
2) составляется стрелочный граф проекта (с добавлением необходимых фиктивных операций);
3) оценивается полезность от сокращения времени каждой из операций;
4) производится сокращение времени части операций и рассчитывается новое время выполнения проекта и дополнительные затраты.

Постановка задачи

В рамках данной работы требуется разработать программное средство, позволяющее проводить автоматизированный сетевой анализ проектов.

В данной задаче проект рассматривается как совокупность большого числа операций, зависящих друг от друга. Проект задается в виде технологической карты. Некоторые операции могут выполняться параллельно, некоторые лишь после окончания выполнения предыдущей операции. У каждой операции есть время выполнения и стоимость, проект так же имеет ограниченное время выполнения и финансирования. Используется ряд допущений:

- необходимые для выполнения операций детали всегда имеются на складе и вовремя доставляются к началу операции;
- время выполнения операции зависит от одного ресурса (денег);
- время выполнения операции задается детерминировано.

Необходимо оценить общее время выполнения проекта, используя дополнительное финансирование оптимизировать время выполнения проекта или суммарное время простоя каждой операции.

Программное средство должно предоставлять следующие возможности:

- построение стрелочного графа проекта;
- добавление фиктивных операций;
- вычисление критического пути проекта и выделение его на графе;
- минимизация времени выполнения проекта;
- минимизация времени простоя проекта.

Реализация

При написании этой программы был написан новый алгоритм минимизации времени выполнения проекта, основываясь на понятии критического пути. В дальнейшем алгоритм был модифицирован с целью использования его для минимизации времени простоя проекта.

1.1. Минимизация времени выполнения проекта

Алгоритм заключается в следующем.

Вход: список всей путей от начального до конечного состояния AllPath, у операций заполнены поля time (время выполнения), probtime (возможное время сокращения), cost (цена сокращения на день).

Выход: операции с минимальным временем выполнения и затраты на эту оптимизацию.

fin = 0 {показывает изменился ли критический путь проекта}

ttime = NewTime {для хранения последнего оптимального состояния}

repeat

flag = 0 {показывает было ли изменение какой-либо операции}

critpath := MaxPath(AllPath) {длина критического пути}

predcritpath := PrevMaxPath(AllPath) {длина пути, по величине следующего за критическим}

min = ∞

```
for path∈AllPath do
        for op∈path do
      if op.probtime ≠ 0 &  Cost(op.cost) < min  then {Cost
        вычисляет стоимость с учетом кратности операции}
        min = Cost(op.cost);
          x = op {самая дешевая для сокращения операция}
          end if
        end for
  end for
if critpath ≠ fin then{сохраняем состояние}
   fin = critpath
        Save(ttime)
   end if
tmp = 0 {время сокращения до изменения критического пути}
if  length(AllPath) or predcritpath = 0 then
        tmp = 1
else
   tmp = critpath – predcritpath
   end if
if x ≠ nil then
        while (InBudget & x.probtime≠0 & tmp≠0) do
      flag = 1
```

```
            tmp := tmp - 1

            x.time := x.time - 1

            x.probtime := x.probtime - 1

        end while

    end if

until flag = 0

Load(ttime) {загрузка последнего оптимального состояния}

Cost = ProjCost {стоимость сокращения времени}
```

Пояснения:

1. Cost - учитывается кратность операции, т.е. операция, лежащая одновременно на нескольких критических путях, получает стоимость, равную реальной стоимости деленной на число критических путей, в которые она входит.
2. Проверяется (в случае нескольких критических путей), было ли превышение бюджета в середине (то есть критическая операция хотя бы на одном из критических путей уже сокращена, но при этом хотя бы один критический путь не был сокращен) сокращения критического пути. Если да то состояние операций откатывается на состояние до попытки сокращения, т.к. затраты на сокращение в данном случае будут бесполезными (не приведут к сокращению времени проекта – а лишь уменьшат число критических путей).

1.2. Минимизация времени простоя проекта

Алгоритм заключается в следующем.

Вход: список всей путей от начального до конечного состояния AllPath, у операций заполнены поля time (время выполнения), probtime (возможное время сокращения), cost (цена сокращения на день).

Выход: операции с оптимальным временем выполнения и затраты на эту оптимизацию.

```
fin = 0
```

```
ttime = NewTime
repeat
    flag = 0;
    critpath := MaxPath(AllPath)
    predcritpath := PrevMaxPath(AllPath)
    min = ∞
    for path∈AllPath do
      for op∈path do
        if op.probtime ≠ 0 &  Cost(op.cost) < min & IsCrit(op)  then
                    {не сокращается операция, принадлежащая
                          всем критическим путям проекта}
          min = Cost(op.cost)
            x = op
          end if
       end for
     end for
    if critpath ≠ fin then
      fin = critpath
          Save(ttime)
     end if
    tmp = 0
    if  length(AllPath) or predcritpath = 0 then
          tmp = 1
```

```
    else
        tmp = critpath – predcritpath
        end if
    if x ≠ nil then
            while (InBudget & x.probtime≠0 & tmp≠0) do
        flag = 1
        tmp := tmp - 1
        x.time := x.time - 1
        x.probtime := x.probtime - 1
        end while
    end if
until flag = 0
Load(ttime)
Cost = ProjCost
```

Отличия от алгоритма оптимизации критического пути:

1. Операция, принадлежащая всем критическим путям проекта, не сокращается, т.к. это сокращение не уменьшит время простоя ни одной из операций;
2. Не возникает отката из-за превышения бюджета в середине сокращения критического пути, т.к. всё равно происходит сокращение времени простоя.

Заключение

В ходе разработки программного средства, позволяющего проводить автоматизированный сетевой анализ проектов, были написаны алгоритмы минимизации времени проекта и минимизации времени простоя проекта. В дальнейшем планируется расширить эти алгоритмы, внеся нечеткость во время выполнения каждой конкретной операции, так

как программ с подобными функциями оптимизации времени и простоя в нечетких числах не существует.

Список литературы

1. Балашов В.Г. Механизмы управления организационными проектами / В.Г. Балашов, А.Ю. Заложнев, А.А. Иващенко, Д.А. Новиков // М.: ИПУ РАН, 2003. - 84 с.
2. Эддоус М. Методы принятия решений / М. Эддоус, Р. Стэнсфилд // М.: Аудит, ЮНИТИ, 1997. - 590 с.
3. Таха Х.А. Введение в исследование операций: Пер. с англ / Х.А. Таха // М.: Издательство «Вильямс», 2005. - 912 с.

РАЗРАБОТКА ЭРГОНОМИЧНОГО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА НА ОСНОВЕ МЕТОДА АНАЛИЗА ИЕРАРХИИ (НА ПРИМЕРЕ СРЕДЫ MATLAB)

студ. О.В. Щеголева,

асс. Е.Ю. Митрофанова

При разработке программного обеспечения, а именно прикладных программ, программист должен учитывать пользовательские интересы. Тем не менее, зачастую разработчик пишет программу, не учитывая интересы пользователя. Код программы может быть достаточно эффективным с точки зрения времени выполнения исполняемого кода и занимаемого объема памяти, но эти характеристики будут интересовать только программистов, потому что будут понятны только им и только ими будут оценены должным образом.

К программе предъявляются требования не такие, какими видит их разработчик. И, соответственно при выборе того или иного программного продукта, пользователь будет принимать решение в соответствии с той частью программы, которую он может оценить, а именно интерфейс. Таким образом, главное, что необходимо пользователю для работы с программой - это простой и удобный интерфейс.

Для обеспечения эффективной работы необходимо учитывать:

- эмоциональное состояние пользователя;
- психологическое состояние пользователя;
- физиологические особенности.

Пользовательский интерфейс - разновидность интерфейсов, в котором одна сторона представлена человеком (пользователем), другая -

машиной/устройством. Он представляет собой совокупность средств и методов, при помощи которых пользователь взаимодействует с множеством различных, чаще всего сложных, элементов, машин и устройств.

При проектировании интерфейсов приходится учитывать не столько физические ограничения, сколько психологические, которые часто не принимаются во внимание. Разработчик интерфейса, применяющий психологические знания, может сделать удобной работу совершенно незнакомых ему пользователей, создавая интерфейс, удовлетворяющий не только требованиям с точки зрения вычислительных задач, но и удобный с точки зрения физических и психологических потребностей пользователя.

Таким образом, интерфейс пользователя - это отдельный и очень важный компонент любой системы, требующий предварительного проектирования. Для того чтобы добиться этого необходимо при разработке интерфейса учитывать эргономические аспекты [1].

Эргономика предъявляет определенные требования к взаимодействию человека и машины. Эффективность функционирования системы в целом определяется тем, в какой степени при ее создании были выявлены и учтены присущие человеку и машине особенности, в том числе ограничения и потенциальные возможности.

Эргономичность техники является целостной характеристикой, наиболее обобщенным показателем, венчающим иерархическую структуру свойств и показателей более низких уровней. Эта целостная характеристика вырастает из ряда эргономических свойств техники, которые реализуются при разработке интерфейса: управляемость, усвояемость и «обитаемость».

Существуют некоторые принципы, являющиеся фундаментальными при разработке и реализации различных эффективных интерфейсов.

1. Эффективные интерфейсы должны быть очевидными и внушать своему пользователю чувство контроля. Необходимо, чтобы пользователь мог одним взглядом окинуть весь спектр своих возможностей, понять, как достичь своих целей и выполнить работу.

2. Эффективные интерфейсы не должны беспокоить пользователя внутренним взаимодействием с системой. Необходимо бережное и непрерывное сохранение работы, с предоставлением пользователю возможности отменить любого действие в любое время.

3. Эффективные приложения должны выполнять максимум работы, требуя при этом минимум информации от пользователя.

В соответствии с перечисленными принципами определяются следующие критерии эффективности интерфейсов:

- обучаемость пользователя;
- использование метафор;
- удобная компоновка элементов интерфейса;
- окна диалогов;
- применяемая терминология.

Под пользовательским интерфейсом программы будем понимать совокупность элементов, позволяющих пользователю программы управлять ее работой и получать требуемые результаты. Фактически пользовательский интерфейс - это канал, по которому осуществляется взаимодействие пользователя и программы.

Существуют некоторые принципы, являющиеся фундаментальными при разработке и реализации различных эффективных пользовательских интерфейсов (традиционных графических, так и Web-интерфейсов).

Выделяют следующие принципы: эффективные интерфейсы должны быть очевидными и внушать своему пользователю чувство контроля. Необходимо, чтобы пользователь мог одним взглядом окинуть весь спектр своих возможностей, понять, как достичь своих целей и выполнить работу; эффективные интерфейсы не должны беспокоить пользователя внутренним взаимодействием с системой. Необходимо бережное и непрерывное сохранение работы, с предоставлением пользователю возможности отменить любого действие в любое время; эффективные приложения должны выполнять максимум работы, требуя при этом минимум информации от пользователя.

Таким образом, приложения должны пытаться предположить то, чего хочет пользователь. Не следует заставлять пользователя заниматься поиском информации или вызовом необходимых средств, желательно предоставлять ему доступ ко всей информации и всем средствам на каждом шаге.

При разработке приложения в среде MATLAB разработчик, используя стандартные элементы управления среды MATLAB, не может создать интерфейс, отвечающий всем эргономическим требования. Поэтому было бы целесообразно при разработке интерфейса использовать некую программу, которая будет оценивать созданный интерфейс с точки зрения эргономики, а так же позволять сравнивать несколько интерфейсов. В случаях проведения анализа такого типа, связанных с принятием решения используются различные методы принятия и синтеза решений.

Задача принятия решений (ЗПР) – одна из самых распространенных в любой предметной области. ЗПР заключается в выборе одной или нескольких лучших альтернатив из некоторого первоначального набора. Для того чтобы сделать такой выбор правильно и

как можно ближе к идеальному результату, необходимо четко определить цель и критерии, по которым будет проводиться оценка набора альтернатив.

Одним из самых известных и удобных методов поиска оптимального решения той или иной задачи является метод анализа иерархии (МАИ).

В соответствии с перечисленными принципами эргономики определяются следующие критерии эффективности интерфейсов, которые станут базовыми для построения МАИ:

- обучаемость пользователя;
- использование метафор;
- удобная компоновка элементов интерфейса;
- окна диалогов;
- применяемая терминология

С учётом всех требований разработана программа, которая на основе МАИ определяет наиболее подходящий интерфейс с точки зрения эргономики. В ходе разработки были выявлены несколько важных этапов:

1) определить набор критериев эргономики, на основе которых будет происходить сравнение интерфейсов;

2) определить количество интерфейсов для сравнения, загрузив или создав их;

3) заполнить матрицы парных сравнений;

4) на основе интегральной оценки получить наиболее подходящий с точки зрения эргономики интерфейс;

5) на каждом этапе обеспечить информационную поддержку пользователя.

В ходе разработки первого этапа в качестве эргономичных критериев взяты базовые (стандартный набор критериев, предложенный приложением) критерии, а так же реализована возможность создания своих критериев с добавление их значений (рис. 1).

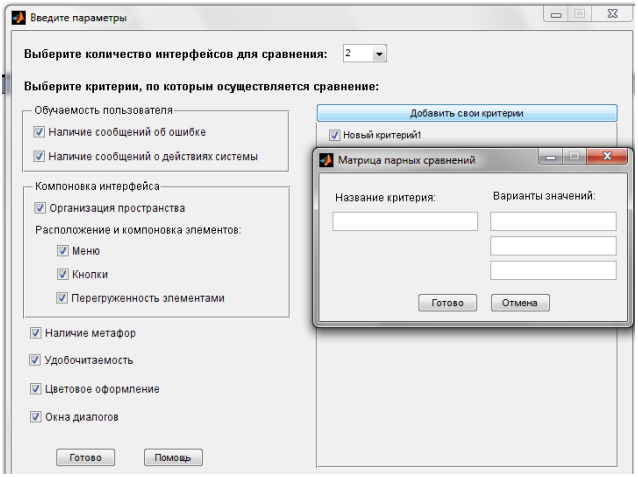

Рис. 1. Выбор и создание критериев

Так же на этом этапе добавлена система помощи. В случае если необходимо разъяснение по каждому из стандартных критериев, то информацию можно получить, щёлкнув правой кнопкой мыши на нужном критерии (рис. 2).

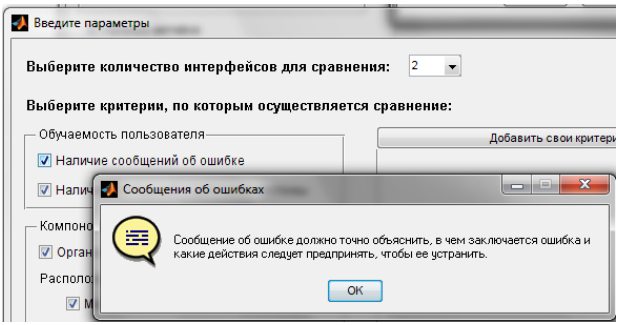

Рис. 2. Справка для стандартных критериев

Определиться с количеством сравниваемых интерфейсов можно на текущей форме (рис.3), для этого необходимо выбрать нужное количество интерфейсов из выпадающего списка (но при этом возможность добавления нового интерфейса также присутствует на следующей форме).

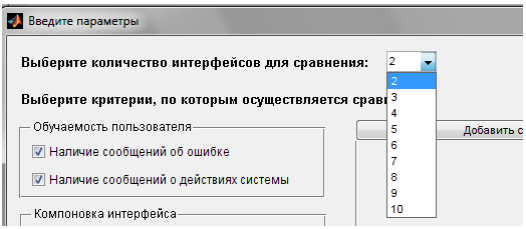

Рис. 3. Количество сравниваемых интерфейсов

На главной форме приложения, которая появляется после выбора критериев, реализована возможность загрузки готовых и создания новых интерфейсов в реальном времени.

В качестве «готового» интерфейса может служить картинка с интерфейсом, либо готовый файл с расширением *.FIG (рис. 4) .

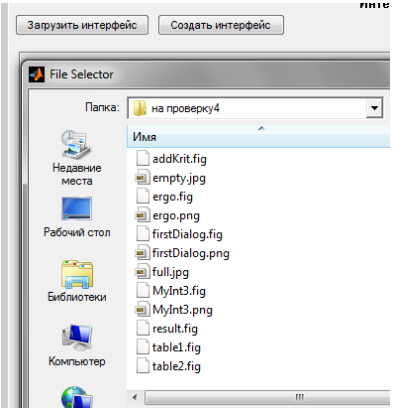

Рис. 4. Загрузка готовых интерфейсов

В случае если пользователь хочет создать свой интерфейс, ему необходимо нажать на кнопку «создать интерфейс» и появится диалоговое окно среды MATLAB, в котором, используя стандартные средства этой среды, можно создать интерфейс (рис. 5).

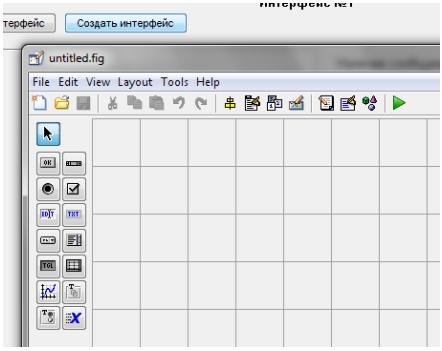

Рис. 5. Создание нового интерфейса

После того как пользователь определился со сравниваемыми интерфейсами и их критериями, ему необходимо заполнить матрицы парных сравнений. Количество матриц определяется по формуле n+1, где n – количество критериев. Чтобы избежать частичного заполнения матриц, для каждой из них добавлены индикаторы, которые позволяют пользователю отследить уровень заполненности матриц (рис. 6).

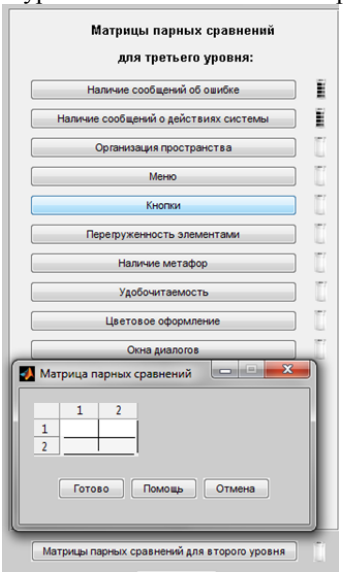

Рис.6. Заполнение матриц парного сравнения

Если все матрицы заполнены, активируется кнопка «рассчитать», при нажатии на которую программа выдаст наиболее подходящий вариант с точки зрения эргономики.

Реализованное приложение, предназначено для нахождения наиболее оптимального интерфейса с точки зрения эргономики в среде MATLAB.

Поскольку используя стандартные средства MATLAB нельзя создать интерфейс, который будет соответствовать всем требованиям эргономики, то явным преимуществом этой программы является, то, что она позволяет определить наиболее эргономичный интерфейс на основе определённых пользователем критериев.

Список литературы

1. Массель Г.Г. Психологические аспекты пользовательского интерфейса современных компьютерных / Г.Г. Массель // Иркутск: ИСЭМ СО РАН, 2000. - 42 с.

2. Головач В.В. Дизайн пользовательского интерфейса v1.2 / В.В. Головач // М.: Usethics, 2000. -143 с.

3. Андрейчиков А.В. Анализ, синтез, планирование решений в экономике: учебник / А.В. Андрейчиков, О.Н. Андрейчикова. – 2–е изд., доп. и перераб. // М.: Финансы и статистика, 2004. – 464 с

СОДЕРЖАНИЕ